

AD-A241 866



RL-TR-91-192
Final Technical Report
August 1991

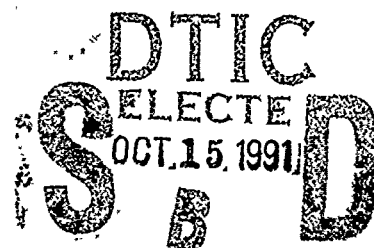


2

OPTICAL CONTENT-ADDRESSABLE MEMORIES FOR DATA/KNOWLEDGE BASE PROCESSING

Georgia Institute of Technology

Thomas K. Gaylord



APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

91-13141



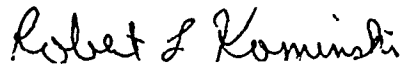
Rome Laboratory
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700

91 1011 017

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

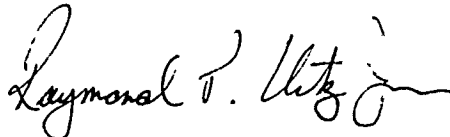
RL-TR-91-192 has been reviewed and is approved for publication.

APPROVED:



ROBERT L. KAMINSKI
Project Engineer

FOR THE COMMANDER:



RAYMOND P. URTZ, JR.
Technical Director
Directorate of Command & Control

If your address has changed or if you wish to be removed from the Rome Laboratory mailing list, or if the addressee is no longer employed by your organization, please notify RL(C3DB) Griffiss AFB NY 13441-5700. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503

1. AGENCY USE ONLY (Leave Blank)		2. REPORT DATE August 1991		3. REPORT TYPE AND DATES COVERED Dec 87 - Dec 88	
4. TITLE AND SUBTITLE OPTICAL CONTENT-ADDRESSABLE MEMORIES FOR DATA/KNOWLEDGE BASE PROCESSING				5. FUNDING NUMBERS C - F30602-81-C-0185 PE - 63728F PR - 2529 TA - 01 WU - PA	
6. AUTHOR(S) Thomas K. Gaylord					
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Georgia Institute of Technology School of Electrical Engineering Atlanta GA 30332				8. PERFORMING ORGANIZATION REPORT NUMBER N/A	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Rome Laboratory (C3DB) Griffiss AFB NY 13441-5700				10. SPONSORING/MONITORING AGENCY REPORT NUMBER RL-TR-91-192	
11. SUPPLEMENTARY NOTES Rome Laboratory Project Engineer: Robert L. Kaminski/C3DB/(315) 330-2925					
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.				12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) This project studied the potential of managing a data/knowledge base using an optical content-addressable memory (OCAM) as opposed to an electronic location-addressable memory. In an optical content-addressable memory, inputs are compared with the stored data, and detected matches found in parallel. In a holographic OCAM, the number of reference patterns represents the number of holograms that need to be stored in the system. Using thick holographic recording media, such as photo-refractive lithium niobate, holograms may be multiplexed together in a common volume. This project completed work on applying OCAMs to the operations of addition and multiplication by using residue-coded numbers and other operations such as discrete matched filtering. These operations can be used to perform data searching or truth-table look-up processing.					
14. SUBJECT TERMS Digital Optical Computing, Optical Content-Addressable Memory, Optical Associative Memory				15. NUMBER OF PAGES 106	
				16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL		

ABSTRACT

In this research we studied the feasibility of implementing and managing a database index scheme using Optical Content Addressable Memories (OCAMs). We review holographic principles, content addressable memory (CAM) and discuss some of the benefits that these fields have for database management. We combine OCAM and database, and indicate how a full set of CAM operations can be performed. Finally, we present a database example to illustrate the ideas.



Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

TABLE OF CONTENTS

	PG.
ABSTRACT	
I. INTRODUCTION	1
II. HOLOGRAPHY	5
1. Introduction	5
2. Basics of Recording a Hologram	6
3. Basics of Reading a Hologram	7
4. Page Composer	8
5. Page Holograms: 2-Dimensional Recording	9
6. Storage and Readout Based on the Exclusive OR Principle	10
7. EXOR Table Look-Up	11
8. Volume Holograms: 3-Dimensional Recording	12
9. Acousto-Optic Deflector (Bragg Cell)	14
10. Magneto-Optic Holography	15
11. Computer Generated Holograms	16
12. Implementation of a Holographic Memory	16
III. CONTENT ADDRESSABLE MEMORY	20
1. Introduction	20
2. Software Implementation: Hash Coding	20
3. Hardware Implementation	22

	PG.
IV. ASSOCIATIVE HOLOGRAPHIC MEMORIES	25
1. Classification of Associative Holographic Memories	25
2. Sakaguchi's and Knight's Holographic Associative Memory	26
3. Fourier Transform Holograms	32
V. OPTICAL CONTENT ADDRESSABLE MEMORIES (OCAMs)	35
1. Introduction	35
2. Equivalence Searches	35
3. Threshold and Double Limit Searches	38
4. Extremum Searches	41
VI. DATABASE INDEX USING OCAMs	45
1. Introduction	45
2. Associative Searches	47
3. Conjunctive Query	49
VII. CONCLUSION	52
VIII. REFERENCES	53
IX. APPENDIX: PAPERS WRITTEN	55
1. The Management of Large Indexes Using Optical Content Addressable Memories	56
2. Optical Content Addressable Memories for Managing an Index to a Very Large Data/Knowledge Base	82

LIST OF FIGURES

	PG.
1. A Comparison of Various Memory Types in Terms of Access Time and Storage Capacity [Gaylord 79]	2
2. A Comparison of Various Memory Types in Terms of Access Time and Cost Per Bit of Stored Data [Gaylord 79]	3
3. Formation of a Hologram	6
4. Generation of a Virtual Image	7
5. Generation of a Real Image	8
6. Page Hologram Array with Page Composer	9
7. Data Representation	10
8. EXOR Table Look-Up Example	12
9. Recording of a Volume Hologram [Collier 71]	13
10. Acousto-Optic Deflector	14
11. Typical Acousto-Optic xy Deflector System [Gaylord 79]	15
12. A Magneto-Optical CAM [Kohonen 79]	16
13. Schematic of Holoscan [Sutherlin 74]	18
14. Hash Data Table	21
15. Basic Associative Memory Operations	23
16. A CAM Organization	24

	PG.
17. Classification of Optical Associative Memories [Paek 87]	25
18. Hologram Construction [Knight 74]	26
19. Sakaguchi's Associative Holographic Memory [Sakaguchi 70]	27
20. Schematic of Sakaguchi's Configuration [Sakaguchi 70]	28
21. Real Image Reconstruction in Parallel Readout [Knight 74]	30
22. Integration of Page Energies in Parallel Readout [Knight 74]	30
23. Readout of Selected Hologram Page [Knight 74]	31
24. Schematic Diagram of the Fourier Holographic Memory System [Paek 87]	33
25. The Four Patterns Stored in the Holographic Memory [Paek 87]	34
26. Partial Inputs and the Associative Recalled Outputs [Paek 87]	34
27. Search Arguments and Associative Memory Table	36
28. Results of EXOR Table Look-Up at Photo-Detector Array	37
29. LEG Coding Scheme	38
30. LEG Example	40
31. Summary of Complex Searches	42
32. Maximum Search Example	44
33. Database Excerpt	46
34. Indexes	48

	PG.
35. OCAM Representation	48
36. Conjunctive Query Setup	51
37. Formation of a Hologram	59
38. Reading a Hologram	60
39. Page Hologram with Page Composer	61
40. Data Representation	62
41. EXOR Table Look-Up Example	64
42. Search Arguments and Associative Memory or Table	65
43. Results of EXOR Table Look-Up at Photo-Detector Array	66
44. LEG Coding Scheme	67
45. LEG Example	69
46. Summary of Complex Searches	71
47. Maximum Search Example	73
48. Database Excerpt	75
49. Indexes	76
50. OCAM Representation	76
51. Conjunctive Query Setup	79
52. Formation of a Hologram	84
53. Reading a Hologram	85
54. Page Hologram with Page Composer	85
55. Acousto-Optical Deflector	86
56. Page Addressing Deflector System	87
57. Sakaguchi's Associative Holographic Memory	88
58. Database Excerpt	89
59. Indexes	89
60. OCAM Representation	90

I. INTRODUCTION

From the beginning of time man has stored data in some form or another; whether it be a caveman's drawings, the Roman Empire's census records, or the Internal Revenue Service's records on tax returns. The search for a storage medium is an on going process and along with it, is the search for methods to access the information as quickly as possible. As our society has become more complex, new ways of storing information have been developed, also our ability to search that data has improved. A prime example is the advent of the electronic computer. As the computer moved from the laboratories into the real world the search was on to develop a memory system that would allow the computer to save data and to easily retrieve that data. At first not much information was stored on electronic media due to its expense, but as the cost of the storage media decreased and the storage capacity increased more and more information was stored electronically.

Not only was information stored, but it was retrieved. At first the information was retrieved and searched sequentially for the required data, but this soon became impractical with the advent of large databases. Many accesses had to be made to secondary memory to retrieve the data, and then the data was searched in main memory. Various memory hierarchies were developed to speed up data retrieval. Even then this wasn't enough, because most of the data brought to main memory was not pertinent to the search. Schemes were developed to index the information (group it) in some way so that similar information was held (grouped) together. Various indexing and search schemes were developed to lessen the number of disk accesses to find the desired information. But searching at an index level was still primarily sequential, or some variation of it (binary, block), nevertheless it was not a parallel search.

Associative or Content Addressable memories were introduced to search the data in parallel. These associative memories were/are quite expensive, and therefore only a

small CAM could be included in any system. Since there was only a small amount of such memory in the machine, this meant that the CAM would have to be constantly loaded and reloaded with new data. This led to the realization that even though the searching of the CAM took only a short time, the loading of the CAM caused a bottleneck in the flow of data.

There is a trade-off between access time, memory capacity, and cost per bit; that is, a fast access time is associated with small expensive memories and slow access times are representative of larger, less expensive memories.

Holographic memories offer the possibility of mitigating the effects of these trade-offs. Consider Figure 1 and 2 from [Gaylord 79].

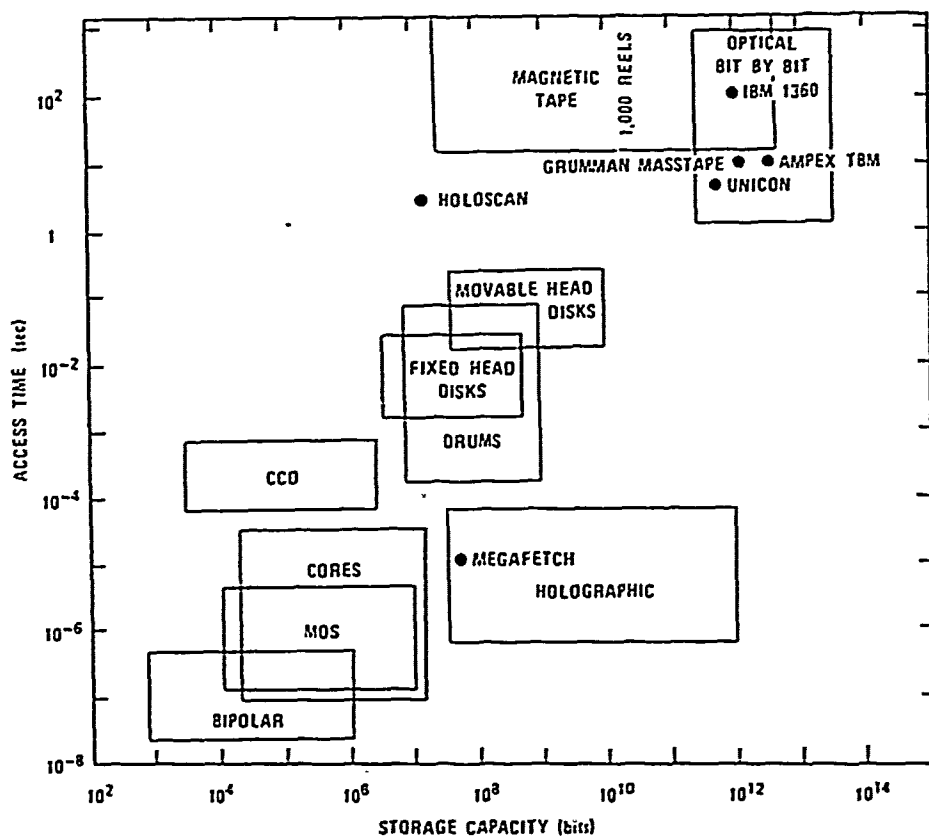


Fig. 1 A comparison of various memory types in terms of access time and storage capacity [Gaylord 79].

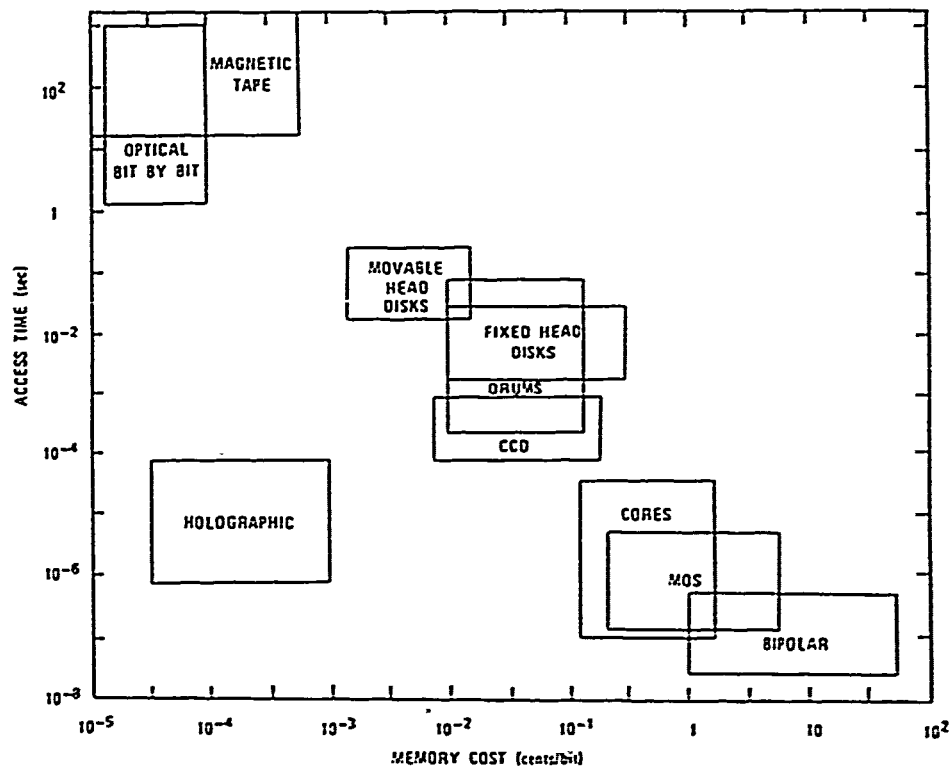


Fig. 2 A comparison of various memory types in terms of access time and cost per bit of stored data [Gaylord 79].

The first graph is a comparison of various memory types in terms of access time and storage capacity. As can be seen from the graph, holographic memories provide a large storage capacity with low access time. Bipolar devices have lower access times, but their storage capacity is much less. Looking at the second graph, which is a comparison of various memory types in terms of access time and cost per bit of stored data we see why holographic memories are so attractive. They are quite inexpensive, in the range of magnetic tape, but offer a much faster access time.

Another benefit of holographic memories is that they are not volatile. They are not easily damaged, due to the way that they are recorded.

Since databases tend to grow, the need to search them also grows. Indexes have been developed to speed up the searching. However, certain types of index structures

offer a fast response time, but the trade-off is that the index itself may be as large or larger than the database. Thus, it is important to consider the storage and access of such indexes using Optical Content Addressable Memories (OCAMs) based on holographic principles.

The first part of this report provides an overview of holography. The second part discusses why the CAM principle is so desirable. In the third section we describe some of the work that has already been done on associative holographic memories. In the next section we provide a method for implementing a full set of content addressable operations on an OCAM. Finally, we provide a database example which illustrates the ideas.

II. HOLOGRAPHY

1. Introduction

The term "holography" was first coined by the inventor of holography, Dennis Gabor. The term is derived from the Greek "holos" meaning "the whole or entirety. A hologram records the whole of the wave information about the wave at each point on a 2-dimensional surface [Gabor 69]. The wavefront can easily be reconstructed so the 3-dimensional information about the wave can be retrieved at will.

In order to see why holography is useful in storing information we draw an analogy between holography and photography, since most people are acquainted with photography, but not necessarily with holography. First, the analogy between holography and photography will be presented, and then a more in depth discussion of holographic principles will be undertaken.

A photograph captures a light image, and when developed we can see the image that was captured by the film. Basically, holography also captures an image, but it captures the waveform of the image. Photography provides a method of recording the 2-dimensional irradiance distribution of an image. Generally speaking each "scene" consists of a large number of reflecting or radiating points of light. The waves from each of these elementary points all contribute to a complete wave, which is called the *object wave*. This complex wave is transformed by the optical lens in such a way that it collapses into an image of the radiating object. It is this image that is recorded on the photographic emulsion (film).

Holography is quite different. With holography, one records the object wave itself and not the optically formed image of the object. This wave is recorded in such a way that a subsequent illumination of this record serves to reconstruct the original object

wave. A visual observation of this reconstructed wavefront then yields a view of the object or scene which is practically indiscernible from the original. It is thus the recording of the object wave itself, rather than an image of the object, which constitutes the basic difference between conventional photography and holography sometimes referred to as "lensless photography" [Smith 69],[Caulfield 70].

2. Basics of Recording a Hologram

A hologram is formed by the interference of two coherent light sources (lasers). The phase information is preserved in two-beam interference patterns. The basic technique of forming a hologram is to first divide the coherent light beam coming from a laser into two beams. Then, one of the beams is used to illuminate the object and the other acts as a reference, and therefore they are referred to as the *object beam* and the *reference beam* respectively.

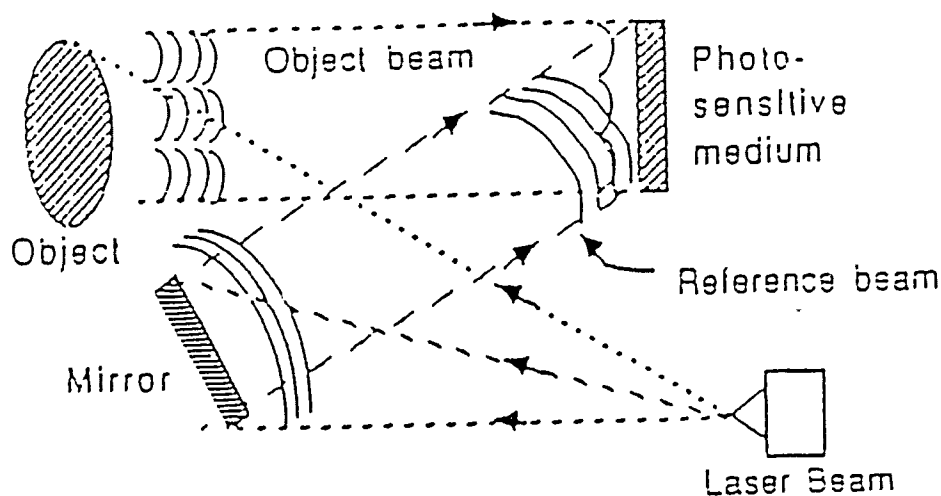


Fig. 3 Formation of a Hologram

The reference beam is directed so that it will intersect and overlap the object beam at a point where a photosensitive medium is placed. The reference beam and the object beam being coherent, will form an interference pattern on the photosensitive medium. After the beams are removed the medium is processed and the interference

pattern formed by the overlapping of the object and reference beams is called a hologram [Collier 79].

Depending on the type of processing used a hologram may be an absorption or a phase hologram. Also, depending on the thickness of the recording medium the hologram may be classified as a *volume hologram*. The significance of a hologram being a volume hologram will be explained later.

3. Basics of Reading a Hologram

Reading the hologram is performed by illuminating the hologram with the original reference beam and reconstructing the object wavefront.

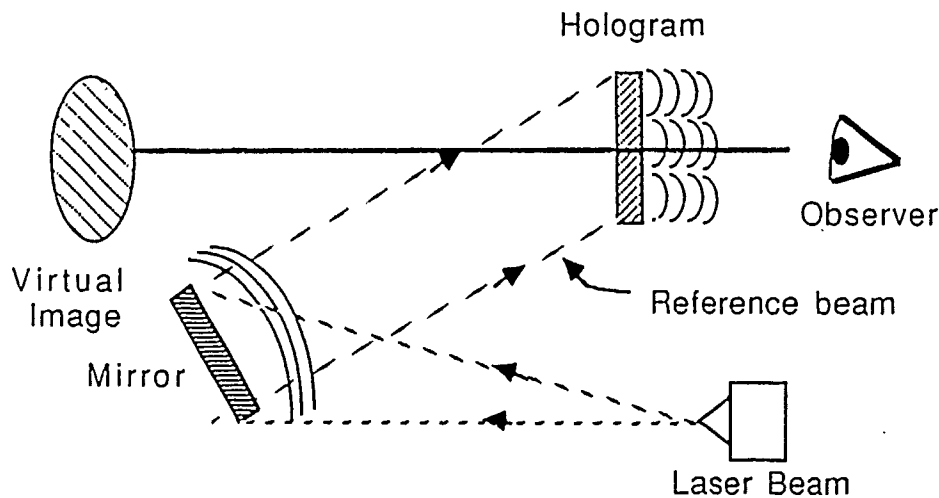


Fig. 4 Generation of a Virtual Image

When the hologram is illuminated by a reference beam, which has the same physical properties as the original reference beam, part of the light diffracted out of the reference beam is directed and shaped by the hologram into a recreation of the light wavefronts originally coming from the object. A reconstructed wave train proceeds out from the hologram exactly as did the original object wave. An observer, viewing a wave identical with the original object wave, quite naturally perceives it to diverge

from a virtual image of the subject located precisely at the original subject location. On the otherhand, if the reference beam is accurately retroflected so that all the rays of the reflected beam are opposite to the original reference, then such a conjugate beam, illuminating the backside of the hologram, produces a *real image* of the subject at the original subject location.

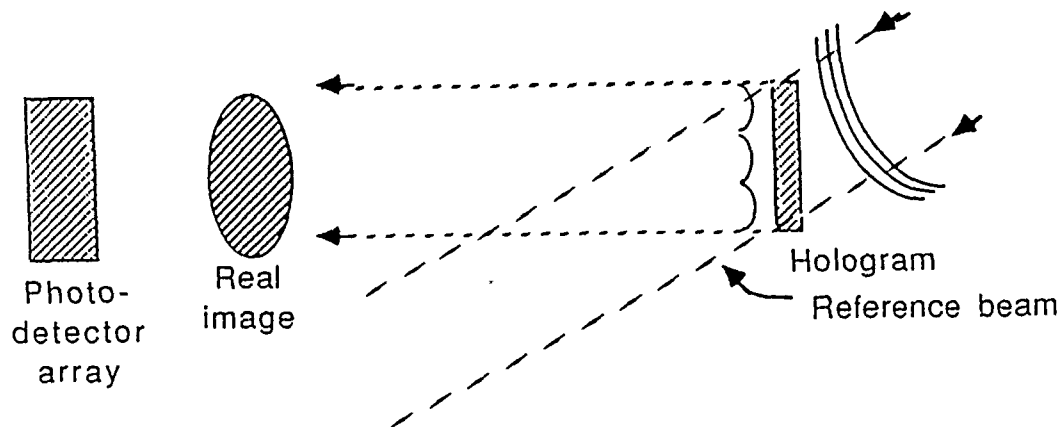


Fig. 5 Generation of a Real Image

Since the light converges to an image, the real image can be directly detected with *photodetectors*, without the need for a lens. Hence, a hologram is a diffracting record of the interference of a particular subject and a particular reference beam.

4. Page Composer

In the process of generating a hologram, the information to be stored is contained in a *page composer*. One particular page composer is called a *latrix* (Light Accessible Transistor Matrix). The *latrix* allows a computer to load data into the memory within the *latrix*. When the memory is full, its contents are recorded into a holographic record at any chosen location.

5. Page Holograms: 2-Dimensional Recording

A *page hologram* array contains many subholograms as shown in Figure 6.

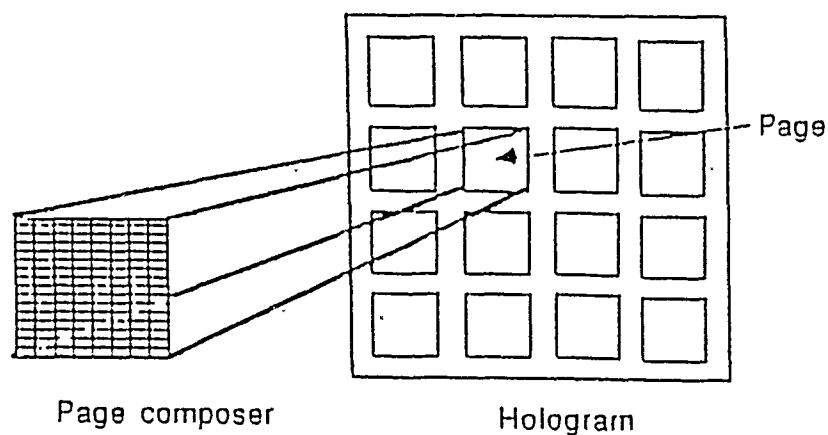


Fig. 6 Page Hologram Array with Page Composer

Each subhologram is called a page. As a page is being recorded in the emulsion, only a small part of the entire hologram is exposed by an aperture to the input. To record the next page, the aperture is moved to a different xy location on the emulsion, and the next page of data is recorded. In the figure there is only one hologram recorded per xy location. However, each page location can be a volume hologram.

Rajchman [1970] was one of the first to explore the concept of page holograms. He defined a page (P) to be a block of bits w by d bits. The number of subholograms is said to be N , where N is the number of subholograms in the array. High bit densities of 10^6 bits/cm² or more can be realized on a page. Having 100 by 100 subholograms, therefore, 10^{10} bits can be stored on a fraction of a square meter, an area small enough to be accessible by a deflected light beam.

Depending on the medium used, information can be erased from a page, and the page can be rewritten with new data.

6. Storage and Readout Based on the Exclusive OR Principle

In this section we explain how data are represented and then recorded in the hologram. Let us say that we have data to record, call it A. The A information is represented by beam A(x), which consists of $2n$ beams, $\bar{a}_1, a_1, \bar{a}_2, a_2, \bar{a}_3, a_3, \dots, \bar{a}_n, a_n$. Each bit of A is represented by a true beam a_i and a complement beam \bar{a}_i , where i represents the i th bit. The light beams that make up A(x) come from the true or complement bit places, according to whether the corresponding bits are in states "1" or "0". Suppose that (A_1, A_2, A_3, A_4) is (1,0,1,0) then, A(x) is a set of beams coming from $a_1, \bar{a}_2, a_3, \bar{a}_4$ that can be used to record the A data, as shown in Figure 7.

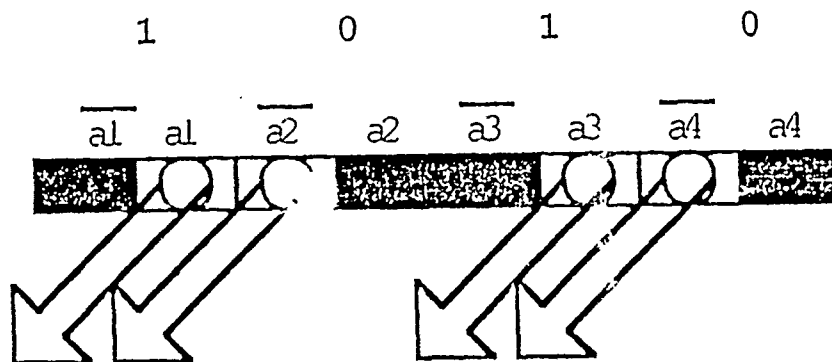


Fig. 7 Data Representation

The holograms are read by interrogating them with a search argument $C=(C_1, C_2, C_3, \dots, C_n)$, represented by beam C(x). Beam C(x) consists of $2n$ light beams, $\bar{c}_1, c_1, \bar{c}_2, c_2, \bar{c}_3, c_3, \dots, \bar{c}_n, c_n$. A match is detected by the exclusive OR principle; that is, a match occurs if the i th interrogation signal bit and the i th interrogated bit are different. So, in actuality the beam that does the interrogation, $C'(x)$, is the complement of beam C(x). The corresponding c_i or \bar{c}_i beams simultaneously illuminate the corresponding a_i or \bar{a}_i bits of all the words subjected to interrogation. Some of the C_i' can be DON'T CARES (dc); as such they do not participate in the interrogation so no light is emitted from the corresponding c_i or \bar{c}_i bit. To illustrate the point the previous value of A is

used, where $A = (1,0,1,0)$. If the search argument (C_1, C_2, C_3, C_4) is $(1,0,dc,0)$, then the corresponding interrogation beam $C'(x)$ is actually $(0,1,dc,1)$; that is, it consists of three parallel beams coming out of \bar{c}_1, c_2 and c_4 positions, with no light coming from c_3 or \bar{c}_3 , because the third bit is a don't care. We find that the interrogation signal $C = (1,0,dc,0)$ matches the interrogated signal $A = (1,0,1,0)$, in reality it is the EXOR between $C' = (0,1,dc,1)$ and $A = (1,0,1,0)$, which determines whether a match has occurred or not. The match is detected by a photodetector array; wherever the detector detects a null then a match has occurred. If all the detectors for a word are detected as null then, the word matches the search argument.

7. EXOR Table Look-Up

In this section we describe an associative system, which is based on table look-up. In the table look-up procedure, given a particular search argument we compare it against reference patterns stored in a hologram. Detection of a match is done using the EXOR principle.

The table look-up method has two components. The first component is the memory composed of holograms, and the second is the page composer. The holograms contain the reference patterns that the search argument will be compared against. A different reference pattern is recorded in each row of the hologram. From now on we will refer to the hologram as the table.

The second component, the page composer, contains the search pattern, which will be compared against the reference patterns stored in the hologram. The search argument is replicated in all the rows of the page composer. When the search is performed each row of the page composer is compared with each row of the table. If a null row is detected by the photo-detector array, then a match has occurred.

To illustrate this let us take a simplified example (Figure 8). For illustration we ignore the actual coding of bits. The table contains the names of the people working in a department. Each name corresponds to a reference pattern. Only the names of people actually working in the department are recorded as reference patterns. Suppose that now we want to find out whether or not a person of a given name, John, is working in the department. John is loaded into the page composer as shown below. The search argument is flashed against the table. If there is a person named John in the department then a null will occur and it will be detected by the photo-detector array. In the example a null is detected in the third row. Since a null row has been detected, the answer is yes; someone with the name of John works in the department.

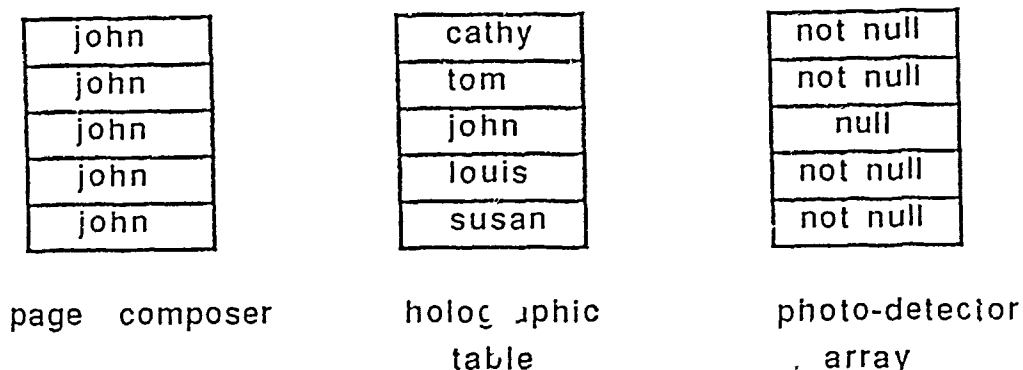


Fig. 8 EXOR Table Look-Up Example

8. Volume Holograms: 3-Dimensional Recording

In a *volume hologram* the thickness of the recording medium is of the order of or greater than the spacing of the vertical fringes. Volume holograms have some very useful and interesting features which are distinct from those of plane holograms. Volume holograms lend themselves very nicely to the storage of information since the whole volume of the recording medium may be utilized.

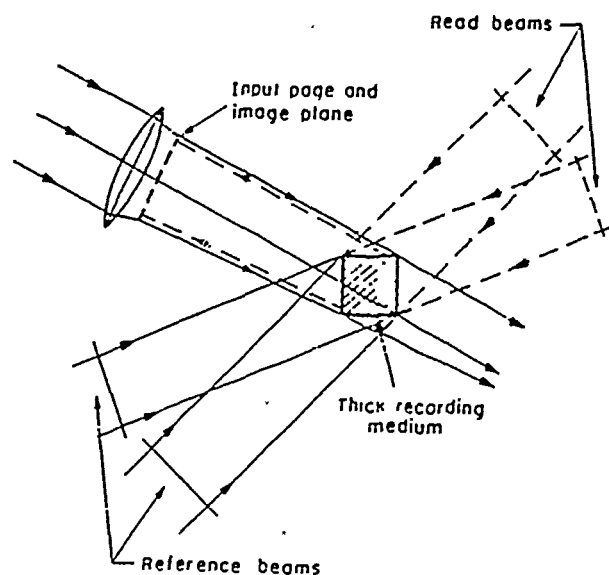


Fig. 9 Recording of a Volume Hologram [Collier 71]

Volume holograms allow the recording of many different holograms in the same emulsion, by just changing the angle of the reference beam (the reference beam's wavelength remains the same), as shown in Figure 9. To reconstruct a particular hologram the reference beam must be deflected to within a narrow angular corridor about the Bragg angle at which that particular hologram was recorded. As the number of holograms recorded in a volume increases, the angular corridor decreases [Gaylord 79]. Illumination outside of this corridor for a particular hologram, causes the intensity of the reconstructed data to be diminished. Beam deflection for these purposes can be accomplished by an acousto-optical modulator, or otherwise known as a Bragg cell. In other words, the holograms are superimposed on each other. As early as 1968 [LaMachia 68] it was shown that it was possible to superimpose 1000 holograms in the same area of photographic emulsion. However, the superposition of more and more holograms in a particular volume, introduces the problem that a hologram being written will affect the holograms already present in the volume.

This particular problem can be reduced by applying an external electrical field to the material (Lithium Niobate) in which the recording is taking place. The application of the electrical field, greatly increases the material's writing sensitivity, while its erasure sensitivity does not change very much. Thus, as a new hologram is written into the volume, only a slight erasure of the other holograms occurs.

9. Acousto-Optic Deflector (Bragg Cell)

A Bragg Cell is an acousto-optical deflector (Figure 10), it allows us to redirect the direction of a light beam passing through it, quickly and efficiently.

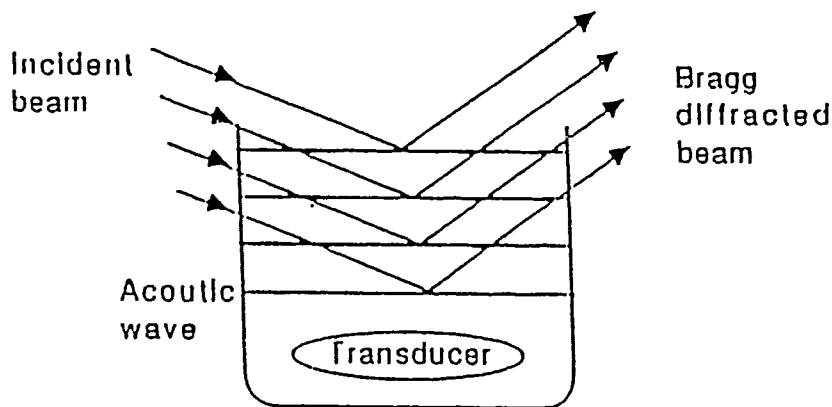


Fig. 10 Acousto-Optic Deflector

At the bottom of the deflector cell there is an acoustic transducer. This transducer introduces an acoustic wave into the elasto-optic medium of the cell. In this way a periodic strain is established in the medium. The cell is now interpreted to be a stack of reflecting layers spaced by the wavelength of the acoustic wave. By turning the acoustic wave on and off, the Bragg Cell can be used to switch directions between the beam which passes straight through the cell, and the beam or beams which are diffracted by the cell.

Usually for holographic applications there will be a deflector for each dimension that the beam must be deflected in. For example, Figure 11 is a xy deflector system. A cascade combination of m deflectors allows 2^m deflection angles [Gaylord 79].

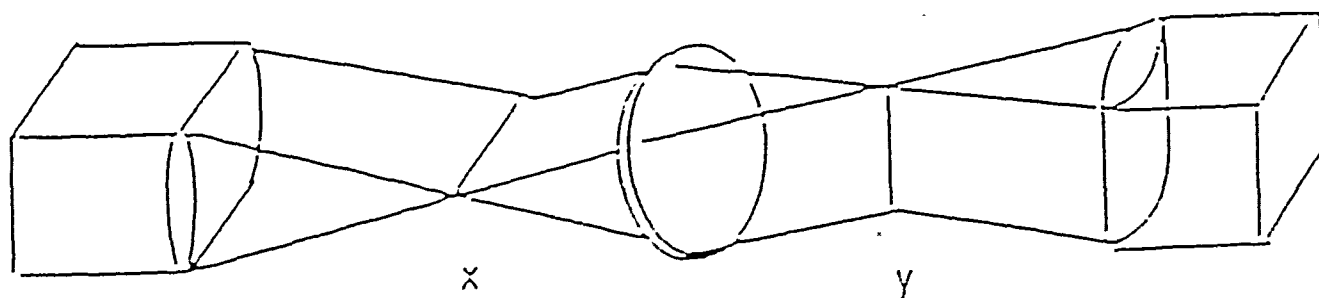


Fig. 11 Typical Acousto-Optic xy Deflector System [Gaylord 79]

Such a deflector can be used in addressing the various pages of a hologram.

10. Magneto-Optic Holograms

Most holograms previously mentioned are write once. A category of holograms that offers possible read write capabilities is magneto-optical holograms. The only difference between these and the holograms we have already described is the recording process. The storage medium is a thin film of MnBi, a magnetic material, that is magnetized normally to the surface into which holographic patterns are recorded by Curie-point writing, and read out is by the Kerr or Faraday effects [Mezrich 70].

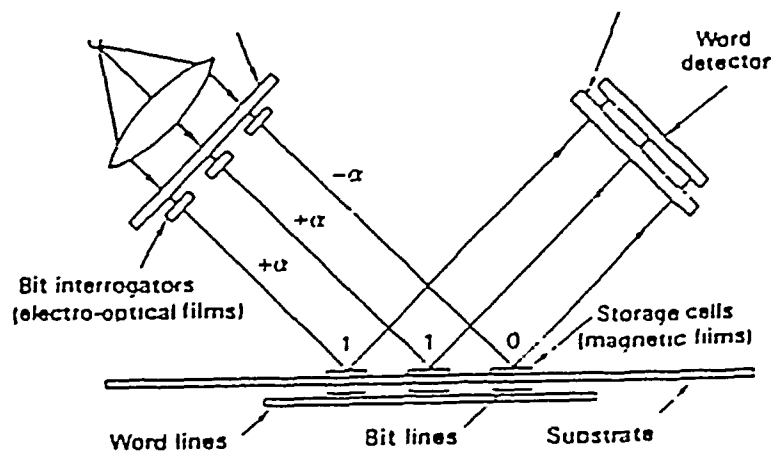


Fig. 12 A Magneto-Optical CAM [Kohonen 79]

11. Computer Generated Holograms

What makes Computer Generated Holograms (CGH) attractive is that a hologram can be made of an object which does not physically exist. Such a hologram can be constructed by calculating the ideal wavefront on the basis of diffraction theory. The field on the hologram surface is represented by a transparency produced by a computer driven plotter, laser beam, or electron beam on correspondingly diverse materials.

12. Implementation of a Holographic Memory

Holoscan [Sutherland 74] was a small self contained holographic reader using 35mm film as the storage medium of digital data in holographic form. Its purpose was to be an off-line credit card verification device for use in department stores, hotels, or other point of sale locations. The objective was to develop a system that could automatically search a large amount of credit card numbers to determine whether the credit card number entered at the keyboard by the sales clerk was valid or not. The clerk was informed of the cards validity by a panel of lights.

Periodically, the invalid card number database had to be updated. This was done by inserting a new cassette of 35mm film containing the new data. Establishments using this system did not record their own holograms, but subscribed to a service that provided them with the updated cassette.

Technical Aspects:

The storage capacity of the film cassette was 250,000 twelve-digit credit card numbers. This amounted to about 12 million bits of information. The Holoscan was designed to offer an answer in not more than 4 to 5 seconds after entry of the card number by the sales clerk. To meet this time requirement, the average time needed to retrieve any number in the memory could not exceed 2 seconds. Also, a lower limit of 100,000 bits/sec was placed on the transfer rate so that the system would perform substantially better than a comparable magnetic tape system.

Holoscan Operation Principle:

The reading process is divided into two parts. The first involves the selection of a single channel or track out of a possible 40 on the film. The second involves the sequential scanning of the film past the stationary laser.

A moveable mirror is mounted to a lead screw and moved back and forth under closed loop servo control to direct the laser beam to the desired channel on the film. A channel runs parallel to the length of the film strip. Holograms within a channel contain the code number of the channel. Thus, as the channel is scanned the position of the beam is always known.

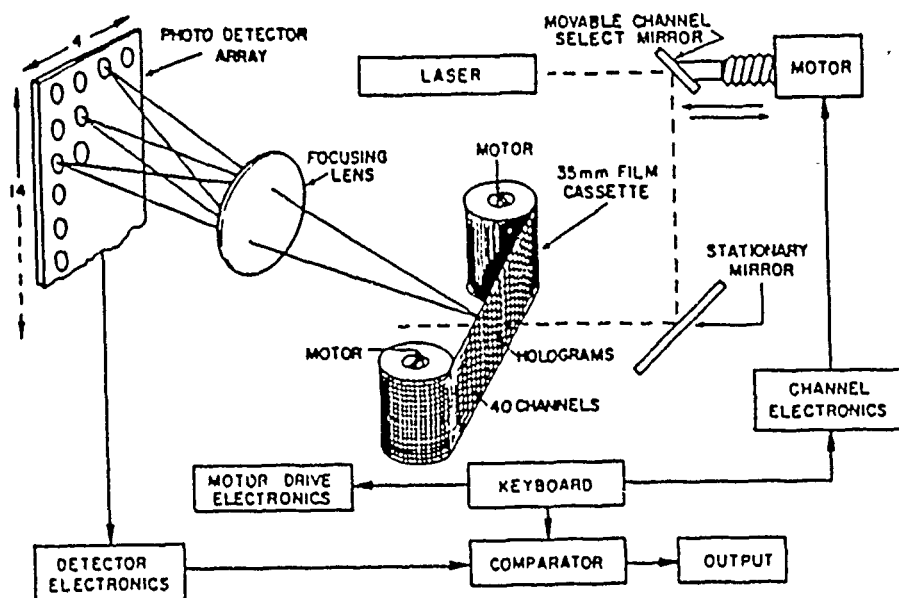


Fig. 13 Schematic of Holoscan [Sutherland 74]

When the correct channel has been found, the beam is locked into position and the film drive mechanism takes over, causing the hologram on the channel to be read out at the rate of 1600 holograms per second.

As each image is reconstructed on the detector array, it is electronically decoded, and scanned in 4 bit parallel fashion in an elapsed time of $150 \mu\text{-seconds}$. A new image appears every $500 \mu\text{-seconds}$ after the scanning operation of the previous one is completed.

As each hologram is readout, the decode account number is compared with the card number entered at the keyboard. If a match is found, then the clerk is informed with a red light. If no match is found the green light comes on. The film pack only holds the invalid card numbers, so when no match is found, then the number entered was valid.

The author goes on to say that a set of such devices can form an archival memory of 10^{12} bits, this would involve 10,000 cassettes of 10^8 bits each. What the Holoscan serves to demonstrate is that holography offers a great amount of storage potential.

III. CONTENT ADDRESSABLE MEMORIES

1. Introduction:

In an extensive review article on Content Addressable Memories (CAMs) by Hanlon in 1966 [Kohonen 79] the following definition is found: "Associative memories have been generally described as a collection or assemblage of elements having data storage capabilities, and which are accessed simultaneously in parallel on the basis of data content rather than by specific address or location."

It is important to note that the accessing of all the data on the basis of their content always means some comparison of an external search argument with part or all the information stored in all cells. Whether this is done by software, mechanical scanning or parallel electronic circuits is immaterial in principle; however, a "genuine" content-addressable memory performs all comparisons in parallel.

Accessing data by Content Addressability can be divided into two categories; one being a software implementation called Hash coding, while the other is a Hardware Implementation called CAM.

2. Software Implementation: Hash Coding

Hashing is based on the principle that if the address space of the memory system were unlimited, the name could be stored at an address which equals its numerical value, and any data associated with the variable would be retrievable in a single access to the memory. However, in real-life the available memory is almost always much smaller than the range of numbers spanned by all permissible names. Because of this some type of compression must take place, but this then leads to difficulties.

NAME	ADDRESS	DATA
John	220	D(1)
Susan	300	D(2)
Herman	182	D(3)
Adolf	126	D(4)
Henry	182	D(5)

Fig. 14 Hash Data Table

Figure 14 contains people's names, the hashed value of the names and the data associated with them. In this case we use only the first two letters of the name to determine the address. We see from the table that Henry and Herman hash to the same value (182), which is called a collision. Since information for both of them cannot be stored in the same location, information about one of them will have to be stored in a different location. There are various collision handling schemes that address this problem. If we cannot store information in the primary location then we must access memory again in order to examine a second location. If this location is already filled, then another location must be found, and this leads to another memory access. This process continues until an empty location is found. To retrieve the information we must follow the same path.

Many schemes have been proposed to reduce the number of possible collisions. The number of accesses is smallest if the items are distributed uniformly in the table. The most important thing to remember is that hashing is not a true Content Addressable Memory, because it does not conduct a true parallel search. Hashing is only a software version of CAMs.

One area in which hashing has been used with success is in database management. Hash coding is most valuable in multiple-keyword searching operations, some

effective implementations based on hash coding can be found in [Kohonen 79].

To expedite the retrieval of information, databases are normally constructed with multilevel storage devices, or in other words there is a hierarchy of memories. The hierarchy ranges from slow access magnetic tape with great storage capacity, to associative memory with low storage capacity, but with fast access time. Even with multilevel storage the entries can be located by hash coding in a time which, in principle at least, is almost independent of the size of memory, and this is an especially important factor with very large databases.

In general, if given a set of identifiers or descriptors, as is the case in searching documents, publications, etc., then hash coding is usually an effective means for performing the task. If on the other hand, the searching conditions are specified by giving the values or ranges of certain attributes, especially using magnitude relations and Boolean functions, then the traditional sorting and tree searching methods would be preferred.

3. Hardware Implementation

A vast majority of all installed computing devices are concerned with pure *searching* and *sorting* tasks, which compromise the majority of administrative data processing. As the data-files increase in size, the time needed to address the records sequentially becomes intolerably long, and therefore, all kinds of organizational solutions and hardware for data management have been suggested and developed to alleviate this deficiency. One of these developments was the Content Addressable Memory (CAM). The CAM is attractive to database applications, because the data can be accessed based on content, and not by addressing. Accessing data by content offers a speed advantage, because the search is conducted in parallel.

The way in which data are specified in a content-addressable search, differs from one application to another. For instance, in certain computations such as text retrieval [Faloutsos 85] it is only necessary to locate entries which exactly match a given search argument. In other database applications, the equality search is not the only type of search performed. There frequently occurs a need to locate all entries which satisfy specified magnitude relations, for instance, having one or several of their attributes above, below, or between specified limits, or absolutely greatest or smallest. Figure 15 shows which type of operations can be performed using an associative memory.

1. Match (equality).
2. Mismatch (not equal to)
3. Less than
4. Less than or equal to
5. Greater than
6. Greater than or equal to
7. Maximum
8. Minimum
9. Between limits
10. Outside of limits
11. Next higher
12. Next lower
13. Various forms of add/subtract

Fig. 15 Basic Associative Memory Operations

Accessing the data on the basis of content always means comparing an external search argument with part or all the information stored in the cells. Figure 16 shows the basic model of a Content Addressable Memory.

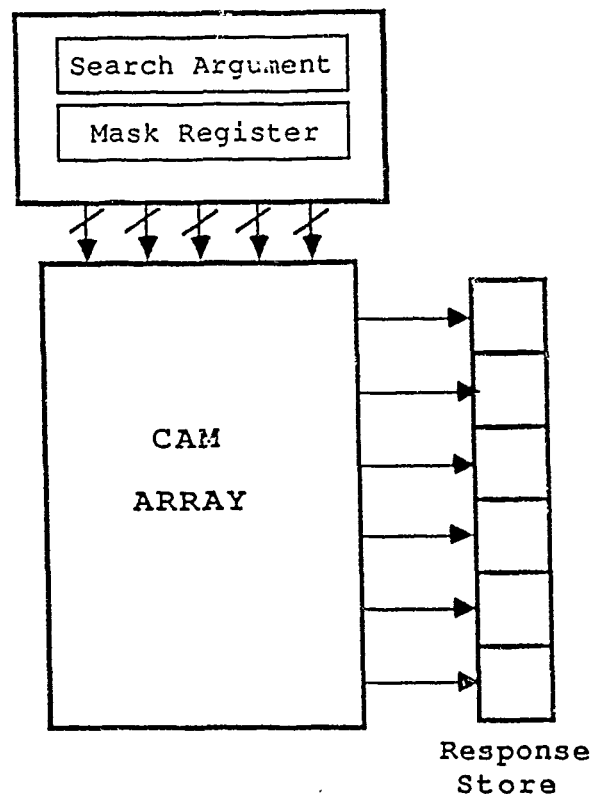


Fig. 16 A CAM Organization

The argument is placed in the *C* register. Register *M* is the mask register and it determines how much of the actual search argument will be used to conduct the search. If all the bits in the *M* register are set to one, then the entire search argument is used to find a match. In this case an exact match is being performed, between the search argument and the data in the array. If only some of the bits are set to one then, the others are referred to as don't care bits.

IV. ASSOCIATIVE HOLOGRAPHIC MEMORIES

1. Classification of Associative Holographic Memories

Optical Associative Memories can be classified into three categories as shown by Figure 17.

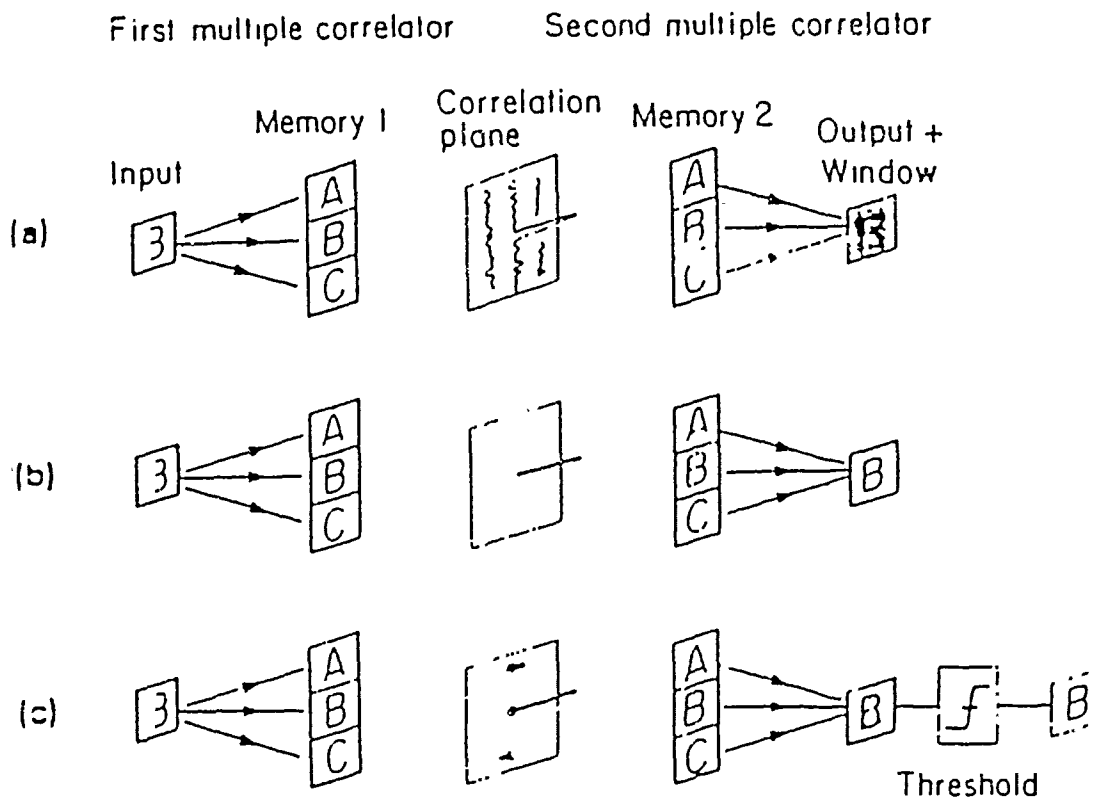


Fig. 17 Classification of Optical Associative Memories [Paek 87]

In the Ghost image memory (a) the input pattern is correlated against all stored information, and the correlation function is then convolved with the associated stored output pattern. The final result is obtained by summing all of the reconstructed images.

In the peak detection method (b) the presence of a peak in the correlation plane determines the best match between the input and one of the stored input images. Once the match has been determined, only that corresponding memory is illuminated.

In the pinhole sampling method (c) the correlation plane is only sampled at the origin where autocorrelation occurs. This spatially sampled correlation peak rather than the entire correlation plane is then convolved with associated stored output image. The number of images stored is not as great as in the other cases, but the quality is much better.

2. Sakaguchi's and Knight's Holographic Associative Memory

Sakaguchi [1970], demonstrated a relatively straight forward extension of conventional holographic storage techniques, that can be used in the implementation of an associative memory. In his paper he discusses a one dimensional associative memory, while Knight [1974] proposes a two-dimensional page storage holographic associative memory based on Sakaguchi's paper.

Sakaguchi's approach is an extension of basic holography. Figure 18 illustrates the idea.

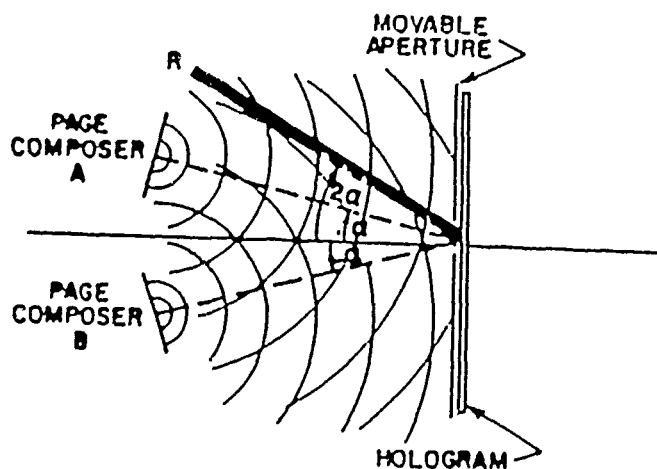


Fig. 18 Hologram Construction [Knight 74]

Instead of a single object beam and a reference beam as in conventional holography, there are two object beams and a reference beam; $A(x)$ incident at angle α to the hologram plane, and $B(x)$ incident at angle β . $A(x)$ represents the search data, i.e., a set

of key words in a library search or some other classification set that ties together common characteristics within the actual data library, $B(x)$.

Sakaguchi uses a thin hologram in his approach. Light beam $A(x,y)$ and $B(x,y)$ contain the information $A = (A_1, A_2, A_3, \dots, A_n)$ and $B = (B_1, B_2, B_3, \dots, B_m)$. Here A_i and B_i denote the i th bit of A and B , respectively. The beam $A(x,y)$ consists of a set of $2n$ parallel beams $\bar{a}_1, a_1, \bar{a}_2, a_2, \bar{a}_3, a_3$ set of $2m$ diverged beams $\bar{b}_1, b_1, \bar{b}_2, b_2, \bar{b}_3, b_3$, A is represented by a true beam a_i and a complement beam \bar{a}_i in the beam $A(x,y)$. The same is true for beam $B(x,y)$. The coherent beams come from the true or complement sources according to whether the corresponding bits are in states "1" or "0". Figure 19 illustrates the idea.

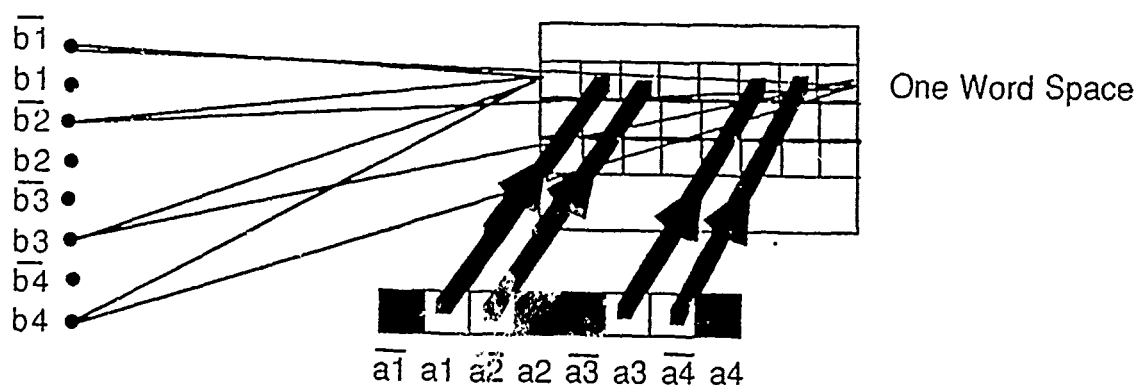


Fig. 19 Sakaguchi's Associative Holographic Memory

Suppose that (A_1, A_2, A_3, A_4) is $(1, 0, 1, 0)$ and (B_1, B_2, B_3, B_4) is $(0, 0, 1, 1)$. Then from Figure 19 $A(x,y)$ is a set of coherent beams coming from $a_1, \bar{a}_2, a_3, \bar{a}_4$, and $B(x,y)$ is a set of diverging beams coming from $\bar{b}_1, \bar{b}_2, b_3, b_4$. In this way the hologram is recorded with A being recorded as a spot sequence \bar{a}_1, a_1, \dots in a word space, while the information B is contained in every spot of $A(x,y)$. The recorded information can therefore be expressed logically by $a_1 \sum (b_i + \bar{b}_i), \bar{a}_1 \sum (\bar{b}_i + b_i) \dots$ etc. [Sakaguchi 70]

Interrogation

Figure 20 shows the configuration of the system that is used for interrogation.

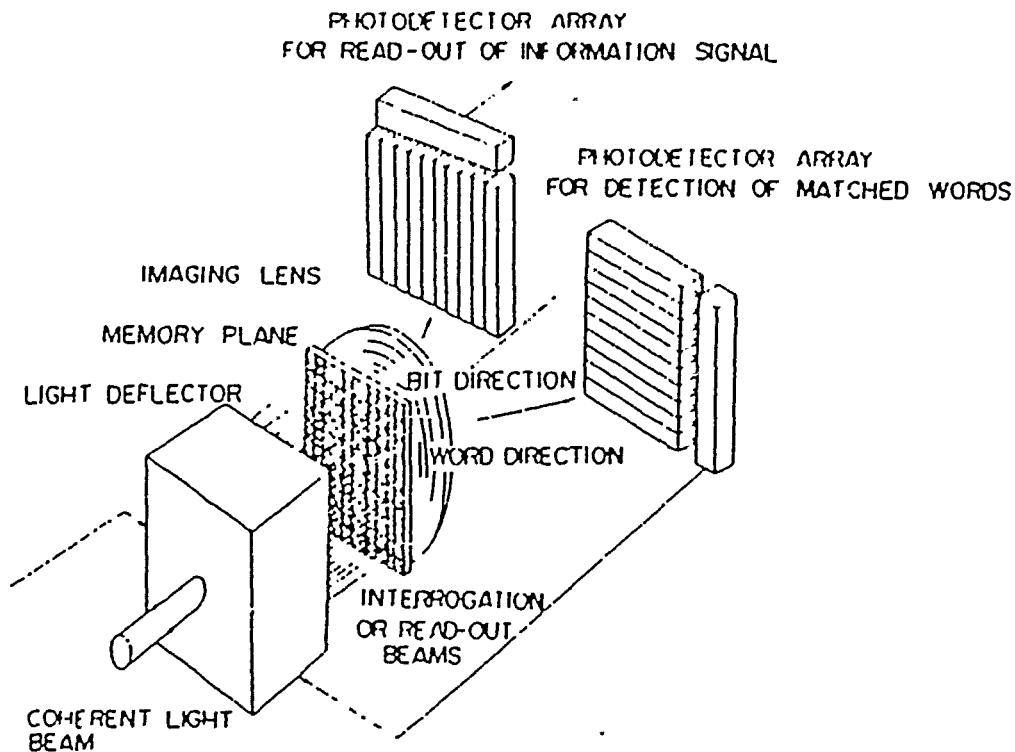


Fig. 20 Schematic of Sakaguchi's Configuration [Sakaguchi 70]

The interrogation is made by the Exclusive Or principle. A coherent interrogation beam $C(x,y)$ modulated by the complement of an interrogation signal $C' = (C'_1, C'_2, C'_3, \dots, C'_n)$ illuminates the memory hologram plane. The beam $C(x,y)$ like $A(x,y)$ and $B(x,y)$ also consists of combinations of true and complement beams; $C(x,y) = \bar{c}_1, c_1, \bar{c}_2, c_2, \dots$ or \bar{c}_i simultaneously illuminates the corresponding bit part a_i or \bar{a}_i of all the words subjected to the interrogation. Some of the C'_i 's can be DON'T CARE bits which do not participate in the interrogation. No light comes from a DON'T CARE bit; for a particular \bar{c}_i or c_i no light is emitted. Since $C(x,y)$ is modulated by the complement of the interrogation signal, no images of the information are reconstructed from the interrogated signals that completely match the interrogation signal. This is the

EXOR principle, a match occurs if the interrogation signal bit and the interrogated bit are different. To illustrate the point the previous example of A and B is used, where $A = (1,0,1,0)$ and $B = (0,0,1,1)$. If the interrogation signal (C'_1, C'_2, C'_3, C'_4) is $(1, X, X, 0)$, where X denotes a DON'T CARE bit. The corresponding interrogation beam $C(x, y)$ is actually $(0, X, X, 1)$; it consists of two parallel beams coming out of the c_1 and c_4 positions. We find that the interrogation signal $C' = (1, X, X, 0)$ matches the interrogated signal $A = (1, 0, 1, 0)$; in reality it is the EXOR between $C = (0, X, X, 1)$ and $A = (1, 0, 1, 0)$. The match is detected by a photodetector array; wherever the detector detects a null then a match has occurred. If all the bits for a word are detected as null then, the word matches the search argument.

This technique was applied by Knight using a page oriented associative memory. Knight describes how to construct a holographic associative memory of 10^8 bits by constructing a matrix of 100 by 100 pages (subholograms) of 1 mm, each having 10^4 bits/page. Each page is recorded with beams $A(x, y)$ and $B(x, y)$ remaining in the same position; only the reference beam is shifted to a new position, and the new area is exposed by opening the aperture. In the meantime, page composers A and B have had new data loaded into them.

In a search the first step is to interrogate the entire memory plane with a coded word (page) A_i and locate a logical match in the output plane. The fact that the entire memory, 10^8 bits can be searched with one flash of light and a match signal derived for each of the pages is important; this is shown in Figures 21 and 22.

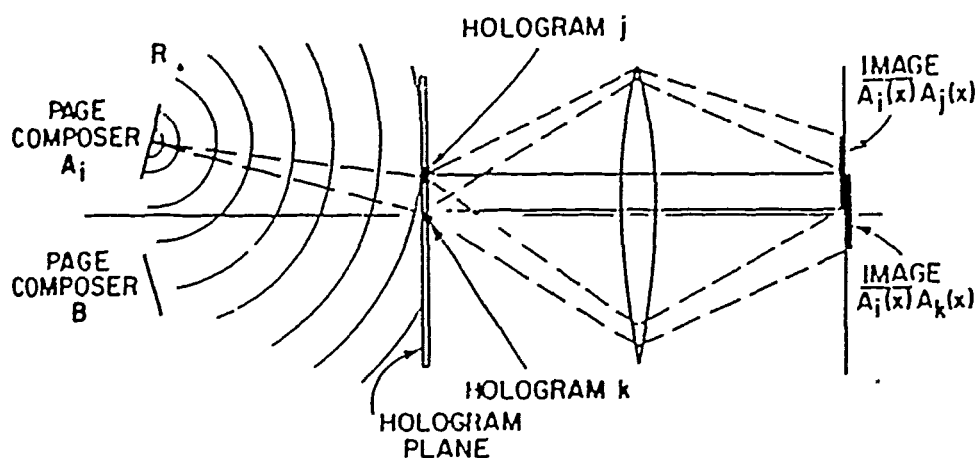


Fig. 21 Real Image Reconstruction in Parallel Readout [Knight 74]

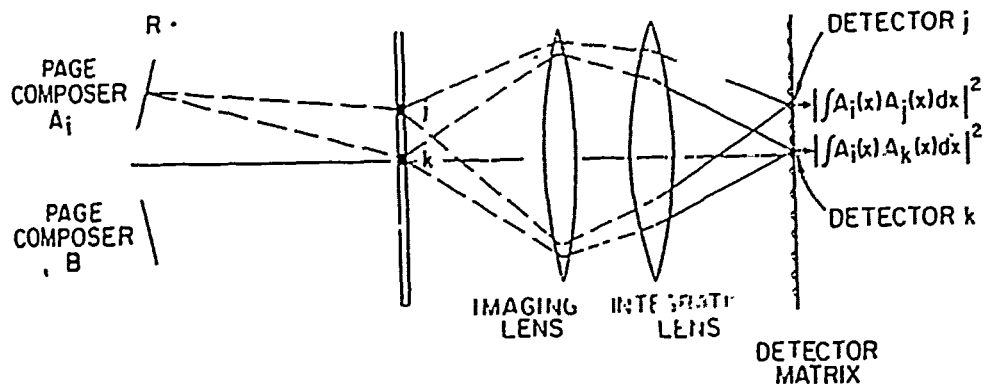


Fig. 22 Integration of Page Energies in Parallel Readout [Knight 74]

A match has occurred when an incident null is detected at the photodetector array. This means that each of those particular pages contains the input word. The derived coordinate is then fed into a two-dimensional acousto-optic deflector, which steers the reference beam in turn to the pages where a match has occurred, Figure 23 illustrates the procedure.

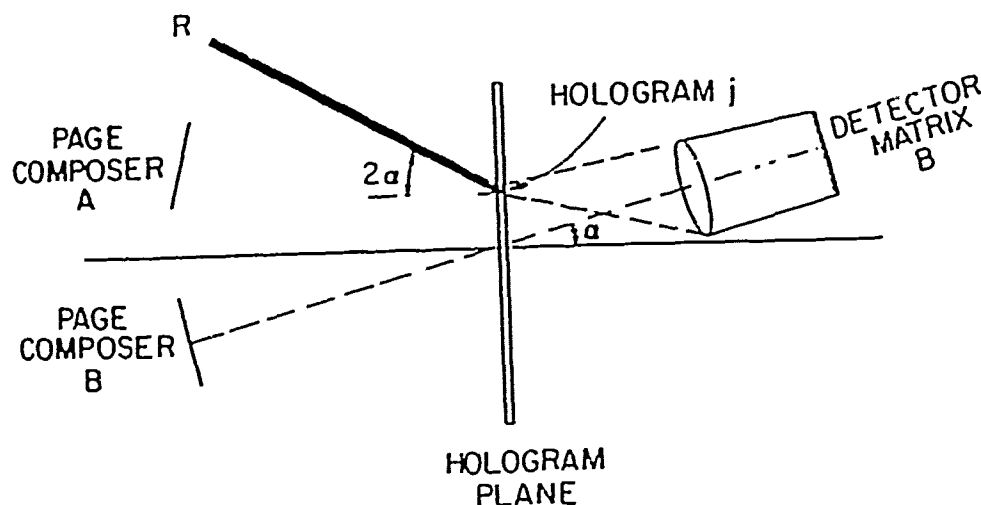


Fig. 23 Readout of Selected Hologram Page [Knight 74]

The page is illuminated thus causing the $B(x)$ information to be imaged onto the detector matrix. The reconstructed data is read electronically by the detector matrix.

Problems

Knight acknowledges that the associative memory is quite different and perhaps less useful than a conventional electronic associative memory. Knight's memory provides for an associative search of 10^4 blocks of information with each data block containing up to 10^4 bits. In a conventional CAM the data is searched in word parallel fashion. What is important here is that an electronic CAM containing 10^5 words, with 100 bits/word would be very expensive to build.

However, there is still a greater problem and that is that the search argument entered in page composer 'A' must be in exactly the same locations as they are stored on the hologram in order to do the match. This problem may be mitigated by using don't care bits in page composer 'A', thus requiring that only a small group of words to be entered into the composer. The question now is what should be done to search for a word, whose location is unknown. In the method described this would involve

strobing the search argument through the page composer until a null is detected at the output. This would be very time consuming, since such a search on a 100 word memory would take on the average 50 searches (strokes). This would take more searches than that proposed by Sakaguchi's one-dimensional associative memory

Optical Correlation

The problem of locating a single word or group of words within a 10^4 bit page, may be solved by optical correlation. In this technique, each individual page must be searched separately [Knight 74]. The complete cross-correlation function of a single hologram in the 10^4 hologram memory plane with the contents of page composer A $[a_j(x)]$ is displayed. Thus, a 40,000 bit detector matrix (for the 200 by 200 bit correlation function) could logically search for a correlation peak.

It is important to remember that if the data to be searched can be kept to a very strict page organization of 10^4 bits, then the previously described method is much more powerful than the correlation method since all 10^4 pages can be searched at the same time.

3. Fourier Transform Holograms

Paek and Psaltis [Paek 87] have suggested another way to implement an associative holographic memory using Fourier Transforms. Figure 24 contains a schematic diagram of their system.

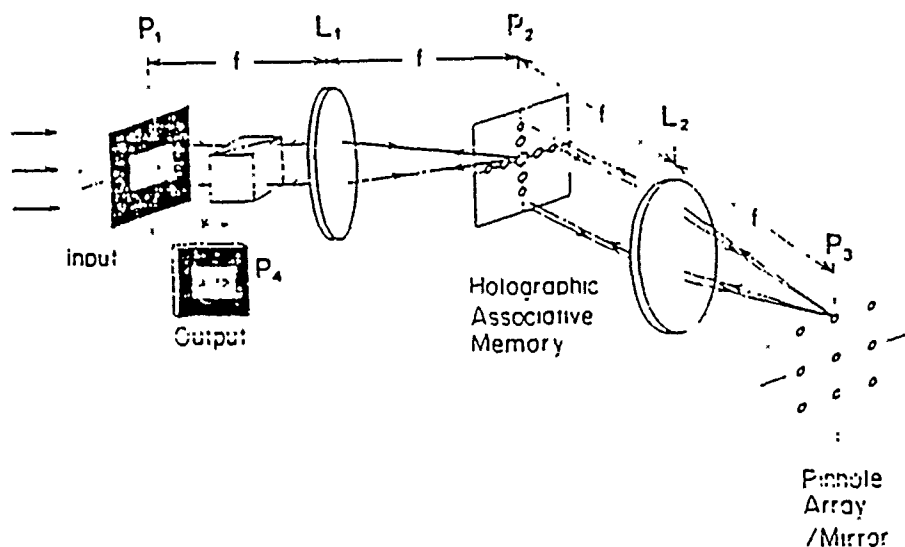


Fig. 24 Schematic Diagram of the Fourier Holographic Memory System [Paek 87]

The input pattern is placed at plane P_1 and is Fourier transformed by lens L_1 . Plane P_2 containing the Fourier hologram is illuminated. The correlations between the input and each memory location are produced at plane P_3 by lens L_2 . The values are sampled by pinholes in P_3 . Each pinhole is located exactly where the stored images were centered when the Fourier hologram was recorded. Light emerging from each pinhole is reflected by a mirror placed immediately behind the pinholes, and the reflected light illuminates the hologram to form the reconstructed images of all the memory locations at the output plane P_4 . The strength at which each memory location is illuminated is proportional to the value of the inner product between the input and the corresponding memory. Some crosstalk is present, since the other memory locations are weakly read out.



Fig. 25 The Four Patterns Stored in the Holographic Memory [Paek 87]

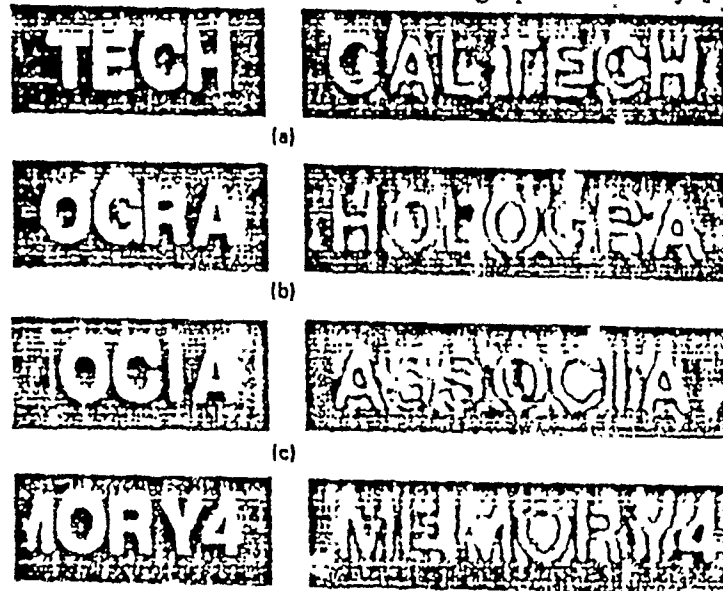


Fig. 26 Partial Inputs and the Associative Recalled Outputs [Paek 87]

Given a partial input, the system can determine the best match to the input and retrieve the entire word. This is illustrated with the simple patterns in Figures 25 and 26.

V. OPTICAL CONTENT ADDRESSABLE MEMORIES (OCAMs)

1. Introduction

In this section we discuss how optics can be used in implementing the various associative memory operations associated with conventional CAMs. Associative operations can be grouped into 4 categories as follows: Equivalence Searches, Threshold Searches, Double Limit Searches, and Extremum Searches. Algorithms to perform these operations in electronic CAMs have been proposed by [Ramamoorthy 78] and [Davis 86].

2. Equivalence Searches

We now demonstrate how equivalent searches can be implemented using OCAMs and the table look-up method. The lower part of Figure 27 represents the words stored in our associative memory or table. For illustrative purposes we limit the number of words to seven of which each word is five bits in length ($n=5$). The light and dark patterns follow Figure 7 and bit values (1 and 0) are included for convenience. Our search argument (S) is 10110 as shown in the upper portion of Figure 27. We can mask (M) any bit as can be done in electronic CAMs by just using DON'T CARES. If we mask bit three, the resultant search argument (S') is 10dc10. Recalling from the EXOR section the actual search argument (SEXOR), is the complemented value as shown in Figure 27. This complemented value is loaded into all the rows (seven in our example) of the page composer.

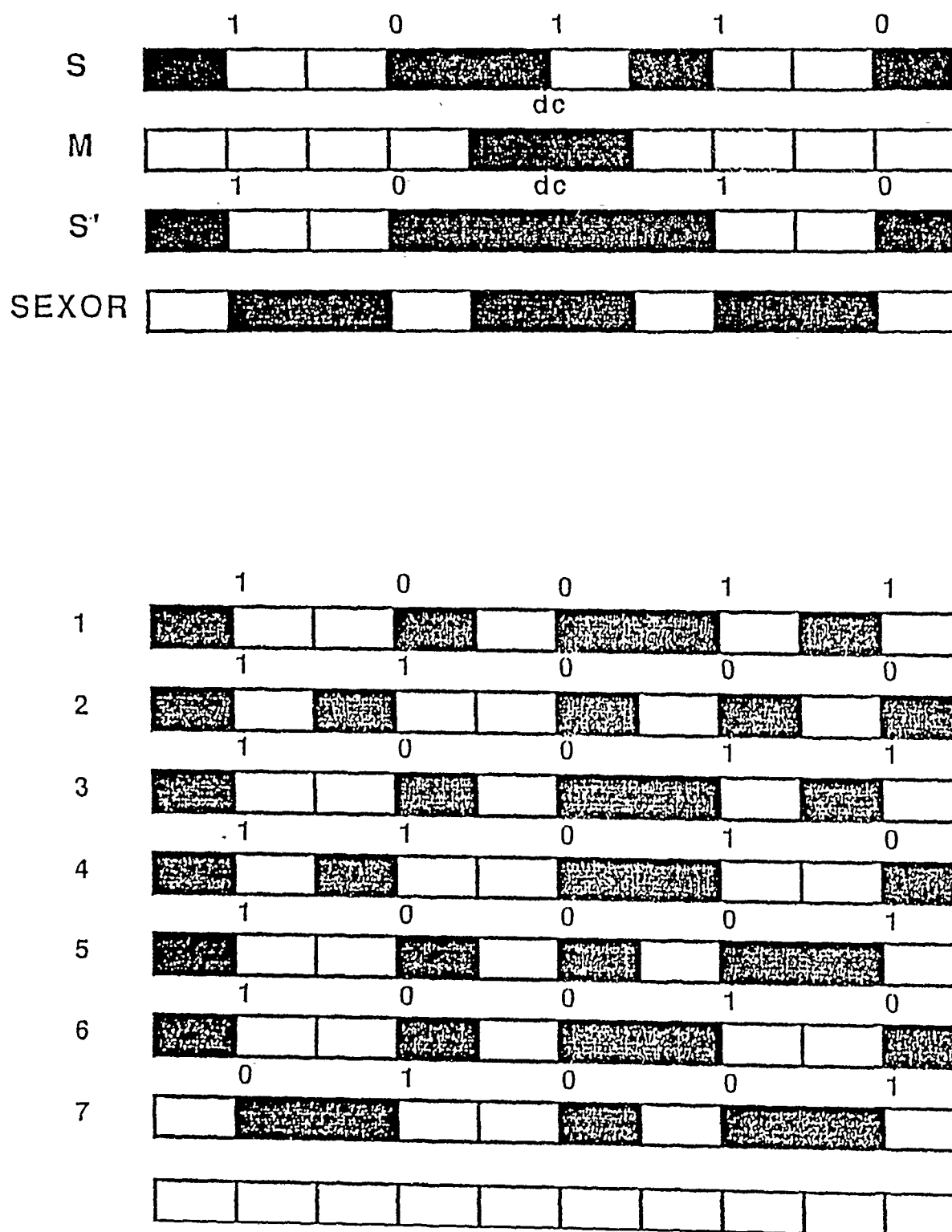


Fig. 27 Search Arguments and Associative Memory Table

The next step is to flash the page composer contents (seven copies of the search argument) against the table. An EXOR is performed between the corresponding bits in the page composer and the table. Figure 28 shows the effect of this operation, as the photo-detector array would see it.

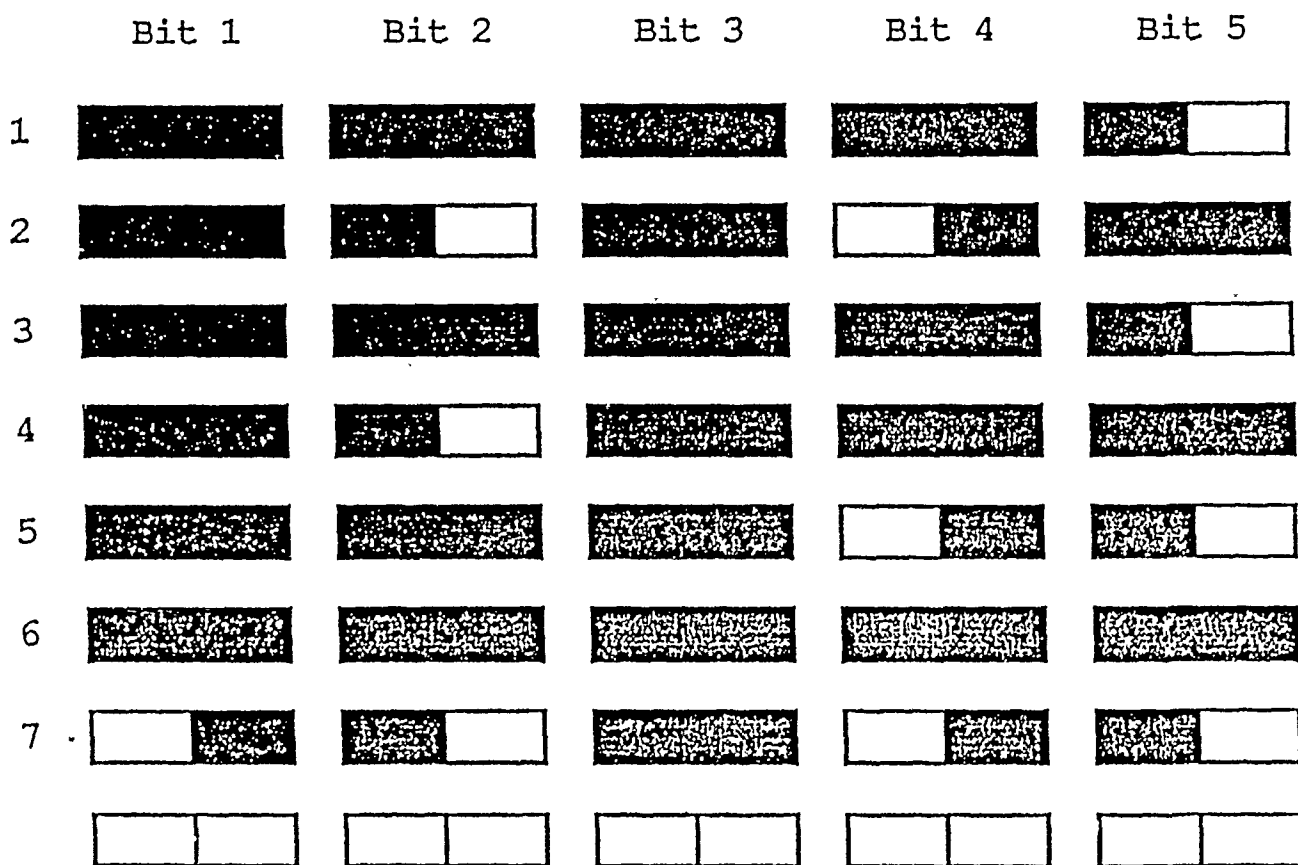


Fig. 28 Results of EXOR Table Look-Up at Photo-Detector Array

As mentioned previously any row containing all nulls is a match. We see that row 6 is a null row, and therefore is equal to the search argument S'. The inequality match is just the opposite interpretation of the equality criteria.

3. Threshold and Double Limit Searches

In this section we show how to perform some of the more complex associative operations. We begin by showing how the words in memory (table) can be divided according to whether they are Less-Than, Equal-To, or Greater-Than the search argument, denoted by L, E, and G respectively. From now on we will refer to this as LEG. Our OCAM approach is based on the LEG concept. Once the search space is divided into LEG the other more complex operations fall right out, since these operations are composed of the unions and intersections of repeated LEG operations.

To divide the search space (SS) into LEG, we will operate in bit serial fashion. The search argument is loaded into the page composer and replicated as before, but instead of flashing all the search argument bits against the table, we start with the most significant bit and operate in bit serial fashion down to the least significant bit. We flash the *i*th bit column of the page composer against the *i*th bit column of the table. The resulting beams are the same as those in Figure 28, but taken a bit column at a time. For convenience we use the notation shown in Figure 29.




		L	E	G
The bit is less than the search argument bit.		1	0	0
The bit is equal to the search argument bit.		0	1	0
The bit is greater than the search argument bit.		0	0	1

Fig. 29 LEG Coding Scheme

The steps to divide the search space into LEG can be summarized by the algorithm below.

Algorithm

n bit word

load the search argument into the page composer and replicate

for i=1 to n do

begin

- flash the ith bit column contents of the page composer against the ith table bit column
- determine any matches between the search and table bits for any row where a null is *not* detected by the photo-detector array do
 - begin
 - interpret the beams striking that photo-detector array row according to LEG coding, and update the LEG Word.
 - disable that particular photo-detector array row.

end

end

collect responses to group the words according to LEG.

We can illustrate this algorithm with an example as indicated in Figure 30. The search argument is loaded into the page composer and replicated. Starting with the most significant bit, the first bit column, we flash it against the most significant bit column of the table. A photo-detector array detects any matches (null rows) and non matches (non-null rows). If a match has occurred no judgement can be made whether the word is LEG than the search argument. However, if a non-null row appears then we can interpret the beams striking the photo-detector array according to the LEG coding scheme (Figure 29). From this information we can determine whether that word is Less-Than or Greater-Than the search argument. As we see in Figure 30, a non-null in the seventh row is detected by the photo-detector array. This non-null is interpreted by the detector array and Word LEG is updated according to the LEG coding scheme as being Less-Than the search argument. The seventh row of the photo-detector array is then disabled, because this non-null contained the information to classify the word

Photo-detector
array

Word LEG

0 1 0
0 1 0
0 1 0
0 1 0
0 1 0
0 1 0
1 0 0

not null

0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
1 0 0

0 1 0
0 0 1
0 1 0
0 0 1
0 1 0
0 1 0

not null

not null

0 0 0
0 0 1
0 0 0
0 0 1
0 0 0
0 0 0
1 0 0

0 1 0
0 1 0
0 1 0
0 1 0

0 0 0
0 0 1
0 0 0
0 0 1
0 0 0
0 0 0
1 0 0

0 1 0
0 1 0
1 0 0
0 1 0

not null

0 0 0
0 0 1
0 0 0
0 0 1
1 0 0
0 0 0
1 0 0

0 0 1
0 0 1
0 1 0

not null

not null

0 0 1
0 0 1
0 0 1
0 0 1
1 0 0
0 0 0
1 0 0

Word LEG

FINAL RESULT

0 0 1
0 0 1
0 0 1
0 0 1
1 0 0
0 1 0
1 0 0

Fig. 30 LEG Example

in row seven as being Less-Than or Greater-Than the search argument.

The second bit column is flashed against the table, but since the seventh row of the photo-detector has been disabled, only the remaining six rows will be detected. We see that there are non-nulls in rows two and four. The beams striking photo-detector array rows two and four are interpreted and the corresponding rows in Word LEG are updated according to the LEG coding scheme. Next the third bit column is flashed, only rows 1, 3, 5, and 6 are detected. Only nulls occur at these rows, as it should be since this bit was a DON'T CARE. We proceed in this fashion until we have looked at all the bits, as seen in Figure 30. The Word LEG is now complete.

At this point we have divided the memory into three sets: Less-Than, Equal-To, or Greater-Than the search argument, L, E, G, respectively. Based on combinations of these sets we can implement the other more complex associative searches, as is summarized in Figure 31. For the Double-Limit Searches, where the limits are denoted by X and Y, two searches must be performed, one for the lower limit, and another search for the upper limit. These searches can also be done in parallel, if provisions are made to include an extra set of page composers and tables. For these searches we have two sets of results, the first set is for the limit X denoted by, LX, EX, and GX. The second set is for the limit Y denoted by, LY, EY, GY.

4. Extremum Searches

In this section we present an algorithm to perform the Extremum searches, maximum and minimum, using OCAMs and the table look-up method. To demonstrate the search we will use the words in memory as shown in Figure 27. Let us first discuss the Maximum search. For this search we use as the search argument, a word consisting of all ones and proceeding in bit serial fashion. We record the maximum value bit by bit in the K register, which is of length n, according to the following algorithm.

I Equivalence Searches

1. Equality Search: detection of null rows
2. Inequality Search: detection of non-null rows

II Threshold Searches

1. Greater-Than Search: G
2. Less-Than Search: L
3. Greater-Than-Or-Equal-To Search: $G \cup E$
4. Less-Than-Or-Equal-To Search: $L \cup E$

III Double Limit Searches

S' is the Search Argument

A. Between-Limits Search, $X < Y$

1. $S' > X$ and $S' < Y$: $LY \cap GX$
2. $S' > X$ and $S' \leq Y$: $GX \cap (EY \cup LY)$
3. $S' \geq X$ and $S' < Y$: $(GX \cup EX) \cap LY$
4. $S' \geq X$ and $S' \leq Y$: $(GX \cup EX) \cap (LY \cup EY)$

B. Outside- Limits Search, $X < Y$

1. $S' < X$ or $S' > Y$: $LX \cup GY$
2. $S' < X$ or $S' \geq Y$: $LX \cup EY \cup GY$
3. $S' \leq X$ or $S' > Y$: $LX \cup EX \cup GY$
4. $S' \leq X$ or $S' \geq Y$: $LX \cup EX \cup EY \cup GY$

Fig. 31 Summary of Complex Searches

Algorithm

n is the bit length of the word

K register of length n to record the maximum word

load all 1's as the search argument into the page composer

for i=1 to n do

begin

- flash the contents of the ith bit column of the page composer against the ith bit table column

- determine any matches between the ith search and table bits
if a null has occurred at any bit row do

begin

- set ith bit in the K register to one.

- disable all the rows of the photo-detector array where a no match (non-null) has been detected (the rest of the rows are still candidates).

end

if no matches (non-nulls) have occurred do

begin

- set the ith bit in the K register to zero.

- do not disable any rows (all rows are still candidates).

end

end

K register contains the maximum word in the search space

Figure 32 shows the determination of the maxima for the example given in Figure 27.

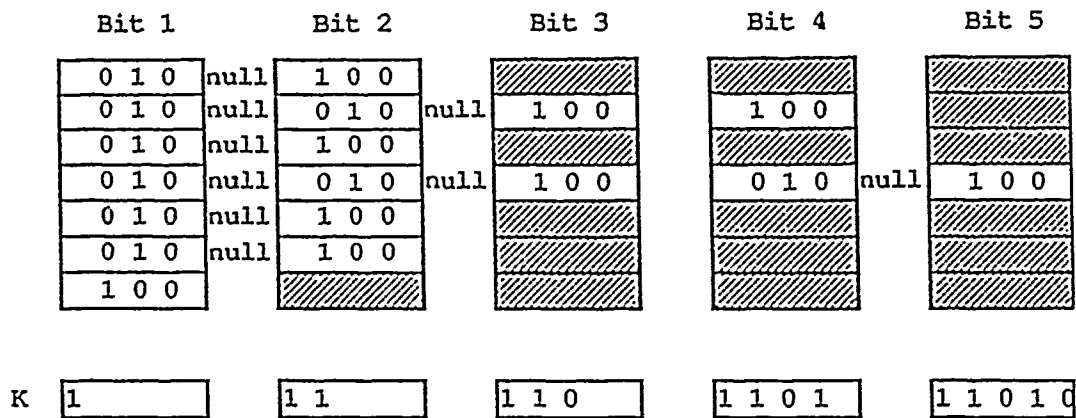


Fig. 32 Maximum Search Example

The algorithm for Minimum search is essentially the same as the Maximum search, except the search argument now consists of all zeroes instead of all ones. When a match does occur for the i th bit, then the i th bit of the K register is set to 0, and is set to one if no matches occur. Once again, at the completion of the algorithm, register K contains the minimum word in the memory space.

Depending on how the associative memory is organized i.e. fields, then we will have to load the maximum (minimum) word into the page composer, and determine its location. Having determined the location we can readout the other fields, which may have been masked out.

VI. DATABASE INDEX USING OCAMs

1. Introduction

In the previous section we have shown how to perform various associative searches on Optical Content Addressable Memories (OCAMs) using a table look-up technique. In this section we present a holographic technique that can be used for managing an index to a very large database. [Sakaguchi 70] demonstrated a relatively straight forward extension of conventional holographic techniques in order to develop a one dimensional optical content addressable memory. Later [Knight 74] refined the idea into a page oriented holographic associative memory. [Berra 88] showed how Sakaguchi's and Knight's associative memory could be applied to managing an index to a database.

Sakaguchi used two object beams $A(x)$, $B(x)$ and one reference beam, instead of one object and one reference beam to record a thin hologram. Information A, is represented by beam $A(x)$, which consists of $2n$ (n is the number of bits) coherent parallel beams, while the B information is represented by beam $B(x)$ consisting of $2m$ (m is the number of bits) diverging beams. The information in beams $A(x)$, and $B(x)$ is represented in binary form as described in the previous section.

Knight (1974) extended Sakaguchi's work by recording a page hologram consisting of 100 by 100 pages each containing 10^4 bits/page, for a total of 10^5 bits. Each page was very small; 1mm^2 . In recording the 10,000 pages the $A(x,y)$ and $B(x,y)$ beams were kept stationary while the reference beam was repositioned for each page. The page composer was loaded with new $A(x,y)$ and $B(x,y)$ data for each page.

What makes this suitable for implementing a database index is that each page has a code word, A, associated with it; and the data, B, written in a particular page can be

identified by using the code word. In other words, the entire memory plane can be searched with a code word represented by beam $A(x,y)$ in parallel with one flash of light. When a match is detected by the photo-array, the data, represented by beam $B(x,y)$, corresponding to that page is read out. Next, an example will be presented that illustrates how this technique can be applied in implementing an index scheme.

In this example we use an inverted list with indirect addressing as the index technique. This particular index was chosen over the more conventional inverted list index, due to the fact that whenever a reorganization of the database occurs (the physical addresses of the records change), the addresses in the holograms do not have to be changed. This is of particular importance since the hologram is read only.

Database						
SS#	Name	Sex	Job	Dept#	Salary	Seniority
012345678	Mary	F	SQ	120	1000	73
123456789	Gary	M	FI	110	2000	64
345678901	John	M	EN	110	1750	76
456789010	Peter	M	JN	130	900	84
567890123	Eva	F	FI	120	2400	79
678901234	Teresa	F	EN	120	1750	81
789012345	Sophie	F	JN	120	1000	83
890123456	Mark	M	SQ	130	1200	77
901234567	Laura	F	FI	110	2200	76
946801234	Hector	M	EN	130	2000	72
956801234	Bruce	M	EN	110	3000	70
968012345	Luke	M	FI	120	2400	74
979123456	Dean	M	SQ	130	1300	74
980123456	Sue	F	EN	120	2000	82
991245678	Cindy	F	JN	130	1100	85
999234567	Anna	F	JN	120	1200	83
999379135	Sam	M	EN	130	2300	80
999387913	Lucy	F	FI	110	2200	80
999581470	Ken	M	SQ	130	1300	72
999826048	Henry	M	FI	110	2400	70

Fig. 33 Database Excerpt

The database shown in Figure 33 is an employee database, and indexes are formed on SS#, Sex, Job, Dept# as shown in Figure 34. We now apply Knight's work to the above database index scheme. The pages in the hologram represent the attributes we have indexed on. For example looking at the Department index we see that there

are 3 different departments (110, 120, 130). With each department number we have associated a list of serial#'s. In recording the data we place the department number on the A(x,y) beam and the list of serial#'s associated with that particular department number on the B(x,y) beam. The representation is stored as shown in Figure 35.

2. Associative Searches

Associative operations can be performed on the keywords, so as to find the page that contains the desired information. For example, suppose we need to find some information about the persons working in Dept 110. To find the page or pages that contain the serial#'s an equivalence search has to be performed between '110' and all the keywords. The interrogation signal is flashed onto the memory plane, and the resultant beams are detected by a photo-detector array. If a null word is detected, then the page corresponding to this keyword position contains the information desired. Having located the page, it is now illuminated and the information is read out in parallel and at the speed of light. The data concerning department 110 may span several pages. In that case each of the pages must be read in turn to get the serial#'s. The serial#'s are now used to obtain the data records. Obviously, a wide variety of query types can be executed and a more complex example is given in the next section.

A1 Index			A2 Index		A3 Index		A4 Index	
Serial#	SS#	Address	Sex	Serial#	Job	Serial#	Dept#	Serial#
1	012345678	32	M	2	SQ	1	110	2
2	123456789	23		3		8		3
3	345678901	14		4		13		9
4	456789010	43		8		19		11
5	567890123	16		10				18
6	678901234	10		11	FI	2		20
7	789012345	6		12		5	120	1
8	890123456	19		13		9		5
9	901234567	22		17		12		6
10	946801234	13		19		18		7
11	956801234	45	F	20		20		12
12	968012345	4		1	EN	3		14
13	979123456	12		5		6	130	16
14	980123456	21		6		10		4
15	991245678	30		7		11		8
16	999234567	36		9		14		10
17	999379135	40		14	JN	17		13
18	999387913	25		15		4		15
19	999581470	18		16		7		17
20	999826048	7		18		15		19
						16		

Fig. 34 Indexes

⁶ 2 18 3 20 110 11	⁷ 1 12 5 14 6 10 7	⁷ 4 15 8 17 10 11 13	
⁴ 1 SQ 19	⁶ 2 18 FI 12	⁶ 3 14 EN 11	⁴ 4 7 11 16 JN
⁸ 2 10 3 1 4 M 8 13	³ 17 19 20 M		
⁸ 1 9 5 14 6 F 7 16	¹ 18 F		

Fig. 35 OCAM Representation

3. Conjunctive Query

In this section we illustrate how a conjunctive query can be performed using the techniques and algorithms described in the previous sections. We use the following procedure

1. Flash in sequence all the search arguments (keys) needed to do the complex query.
 - For each key value note the position and the number of page matches.
2. Order the keys by the number of pages of information they have, least to most.
 - If any have the same number of pages, read the reserved word to determine the amount of data on the page.
3. Read into electronic memory the pages for the key with the least amount of data. This is the source of search arguments for the page composer; the others will be the table(s).
4. Do until all the information from the search argument page is exhausted
 - begin
 - load the search argument values one by one into the page composer.
 - compare the values against the tables (least to most amount of pages).
 - if a match is detected, then proceed to do comparisons with the next table, else no need to go further.
- end
5. If a search argument matches data on all the pages it satisfies the conjunctive query.

To illustrate the above procedure, we perform the following query:

Find all the people who work in Dept 120, who are female (F),
and whose job is finance (FI).

We would in sequence flash these search arguments (keys) against the memory plane such as the one in Figure 35. We would note the position and number of the matches. In our example, for the keys 120, F, and FI, we have 1, 2, and 1 page(s) containing the respective serial# information. We will use the keys which have the largest number of pages as the tables, and the one with the least as the search argument.

The tables will be searched in the order from least to most tables; definitely F will be used as a table, but whether to use FI or 120 as a table could be an arbitrary choice. However, the most time consuming operation is the loading of the page composer with the search argument. For example, if FI had only one entry and 120 had one hundred entries in their respective pages then clearly, it would be desirable to use the information associated with 120 as the table, and the information associated with FI as the source for the search arguments. By choosing 120 as the table we need to only do one page composer load, and a parallel search of all one hundred words is performed, versus one hundred page composer loads and searching only one word at a time if FI was chosen as the table. Therefore, it is beneficial to have the first word in each page be a reserved word, containing the number of serial#'s recorded in that page. In our example FI contains six, while 120 contains seven serial#'s. So the information associated with key FI is read into electronic memory to be used as the source for the search arguments.

Figure 36 illustrates the setup used. To find the answer to the query we proceed as follows. Having read out the information into electronic memory for FI, we take the first serial# and load it into the page composer bank where it is replicated. The number of page composers used in the bank depends on how many pages each table is composed of; one page composer for each page in a table. In our case to do a table

look-up against 120 one page composer needs to be loaded with the search argument. The argument is flashed against the table. In this action we are checking whether the person whose job is FI also works in department 120. If a match does occur, then we proceed to do another table look-up to check whether this same person is female (F). However, if no match had occurred in the 120 table look-up, there would have been no need to do the F table look-up, then the page composer bank would have been reloaded with the next person working in Dept 120. The bank of Bragg Cells steer the search argument light beams coming from the page composer to the proper table. In our example the first serial# from FI to be loaded into the page composer is a '2'. Doing the comparison against table 120, we find that no match occurs, therefore the page composer is loaded with the next search argument, serial# 5. Performing a comparison with serial# 5 against the 120 table, we find that a match occurs. Since a match has occurred we proceed to do a comparison with serial# 5 against the F table. Doing the comparison we find that we have a match, and therefore serial# 5 satisfies our conjunctive query.

The above process could obviously be speeded up by pipelining the table look-ups. Such a speed-up would depend on how many page composer banks the system would be provided with. One set of page composer banks can be doing table look-ups while the banks are being loaded.

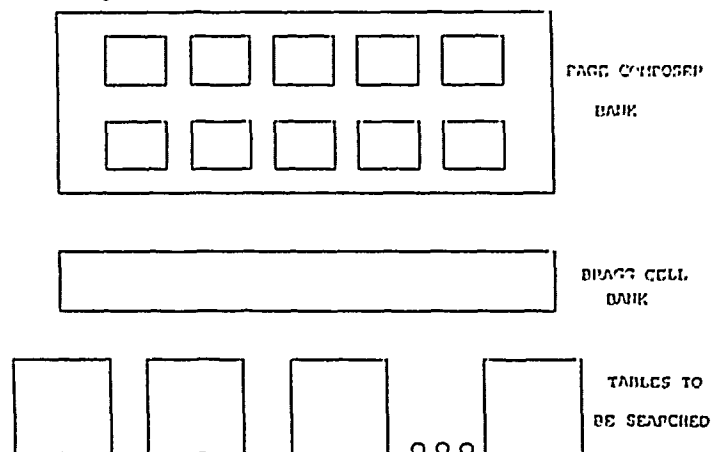


Fig. 36 Conjunctive Query Setup

VII. CONCLUSION

In this report we have introduced the notion of how OCAMs can be used in managing an index to a very large data/knowledge base. Obviously, if the data/knowledge base can fit into the OCAM other approaches can be taken. OCAMs have many attributes, large size, almost instantaneous access and parallel readout, that are important to solving the problems associated with managing an index for very large data/knowledge bases.

The equivalence, threshold, double limit, and extremum searches are well suited for CAM processing. However, due to the expensive nature of electronic CAMs, only a small CAM can be included in any system. The problem with a small CAM for data/knowledge base applications is that it is input output bound. Due to the above detractions the use of electronic CAMs for data/knowledge base applications has not met with a great deal of success.

We present OCAMs as an alternative to electronic CAMs for data/knowledge base applications. OCAMs have the potential for holding megabytes of data at an appreciably lower cost than electronic CAMs. OCAMs perform optically all the various operations associated with conventional CAMs, and in addition they provide parallel output. Holograms do however suffer from the fact that it takes a long time to form them and they are difficult to change once formed. However, for very large data/knowledge bases indexing structures can be devised that are rather insensitive to updates provided that the update rate remains moderate. We are continuing our research into how OCAMs can be applied to other data/knowledge base operations.

VIII. REFERENCES

- [BERRA88] P.B. Berra and S.J. Marcinkowski, "Optical Content Addressable Memories for Managing an Index to a Very Large Data/Knowledge Base", *Data Engineering Bulletin*, March 1988 Vol.11 No.1
- [CAULFIED70] H.J. Caulfield and Sun Lu, *The Applications of Holography*, Wiley-Interscience, Division of John Wiley & Sons, 1970
- [COLLIER71] R.J. Collier, C.B. Burckhardt, L.H. Lin, *Optical Holography*, Academic Press Inc., 1971
- [DAVIS86] Wayne A. Davis, De-Lei Lee, "Fast Search Algorithms for Associative Memories" *IEEE Transactions on Computers*, Vol. C-35, No.5, May 1986
- [FALOUTSOS85] Christos Faloutsos, "Access Methods for Text", *Computing Surveys*, Vol. 17, No. 1, March 1985
- [GABOR69] D. Gabor, "Associative Holographic Memories", *IBM Research Journal*, March 1969
- [GAYLORD79] Thomas K. Gaylord, *Handbook of Optical Holography*, Chapter on Applications, Academic Press Inc. 1979
- [KNIGHT74] Gordon R. Knight, "Page-Oriented Associative Holographic Memory", *Applied Optics*, Vol. 13, No. 4, April 1974
- [KOHONEN79] T. Kohonen, *Content Addressable Memories*, Springer-Verlag Series, 1979
- [LaMACCHIA67] J.T. LaMacchia and D.L. White, "Coded Multiple Exposure Holograms", *Applied Optics*, Vol. 7, No. 1, January 1968.
- [MEZRICH70] R.S. Mezrich, "Magnetic Holography", *Applied Optics*, Vol. 9, No. 10, October 1970

- [PAEK87] E.G Paek, D. Psaltis, " Optical Associative Memory Using Fourier Transform Holograms", Optical Engineering, Vol.26 No.5, May 1987
- [RAJCHMAN70] J.A. Rajchman, "An Optical Read-Write Mass Memory", Applied Optics, Vol. 9, No. 10, October 1970
- [RAMAMOORTHY78] C.V. Ramamoorthy, James L. Turner, Benjamin W. Wah, " A design of a Fast Cellular Associative Memory for Ordered Retieval", IEEE Transactions on Computers, Vol. C-27, No.9, September 1978
- [SAKAGUCHI70] M. Sakaguchi, N. Nishida, T. Nemoto, "A New Associative Memory System Utilizing Holography", IEEE Transactions on Computers, Vol. C-19, No. 12, December 1970
- [SMITH69] Howard M. Smith, *Principles of Holography*, Wiley-Interscience 1969
- [SUTHERLIN74] K.K. Sutherlin, "Holoscan: a Commercial Holographic ROM", Applied Optics, Vol. 13, No. 6, June 1974

IX. APPENDIX: PAPERS WRITTEN

The Management of Large Indexes
Using
Optical Content Addressable Memories

Slawomir J. Marcinkowski and P. Bruce Berra

Dep't of Electrical and Computer Engineering
Syracuse University
Syracuse, NY 13244-1240
U. S. A.

ABSTRACT

In this paper we present an optical approach for managing large indexes. Specifically, our research is focused on managing the indexes using optical content addressable memories (OCAMs) based on holographic principles. The index data are so recorded that associative operations are performed optically using a table look-up method based on the exclusive OR (EXOR) principle. OCAMs offer advantages over conventional CAMs, in that they provide a considerably larger storage capacity (10^8 - 10^{10} bytes) and parallel output. With large capacity, the storing of full index data to very large data/knowledge bases can be achieved, which would otherwise be prohibitively costly in conventional electronic CAMs. The parallel output provides for rapid access to the index data. However, the main disadvantage of OCAMs is that they are read only.

This work was partially supported by the Rome Air Development Center through the Post Doctoral Program at the Georgia Institute of Technology contract number F30602-81-C-0185.

I. INTRODUCTION

One of the major problems in the management of very large data and knowledge bases (10^{12} - 10^{16} Bytes) is the time needed to access the desired data/knowledge. Memory hierarchies exist along with various indexing and search schemes which are designed to minimize the time it takes to access the desired data/knowledge. These techniques have their strengths and weaknesses; so no one method is optimal for all situations. However, since there is generally an increase in performance when using an index over a full sequential search of the entire data/knowledge base all data/knowledge base management systems have the facilities for constructing indexes. While retrieval performance can be improved through the use of indexing it is not without its costs. The main costs are associated with the update of the index data when changes, additions, and deletions are required and in the management of the index data due to its large physical size. In fact, in some applications where considerable index data are required, the size of the indexes will be as large as or even larger than the data/knowledge being managed.

Research over the years has focused on how to improve the access time without incurring too much overhead in terms of update and storage space required for indexing data. The content-addressable memory (CAM) has attracted a great deal of attention for database applications, since data can be accessed by content, and in parallel. This offers considerable speedup since search operations can be performed quickly. The CAM is well suited to perform match (equality), mismatch (not equal to), less than, less than or equal to, greater than, greater than or equal to, maximum, minimum, between limits, outside of limits, and other operations.

However, CAMs do have some disadvantages. The chief constraint on CAMs is that they are quite expensive, and therefore only a small CAM can be included in any system. The problem of a small CAM is that for data/knowledge base applications it is input output bound. The performance of the CAM is governed by the time it takes to

load the data into the CAM, rather than the time to process CAM resident data. In fact, this disparity may be several orders of magnitude. It is for this reason that CAMs have not found widespread use in data/knowledge base systems.

The above comments apply to CAMs that are constructed from electronic components. However, CAMs can be constructed from optical devices, particularly holograms, which are called optical content addressable memories (OCAMs). They have the desirable search processing properties of electronic CAMs yet offer a considerably larger storage capacity (10^8 - 10^{10} bytes) and parallel output. Their main disadvantage is that they are read only. However, if the database under consideration is static and requires fast access then OCAMs offer a potentially cost effective solution.

In our research, we are considering how OCAMs can be used for data/knowledge applications. This paper reports the progress we have made up to this point. The first section provides a brief overview of holography. In the second section we describe how the data are stored in the hologram in order to make use of the exclusive OR principle. The third section describes how the EXOR principle can be used to perform matches in parallel. The fourth section describes how the various associative operations can be performed. Finally, the fifth section discusses how the above can be applied to a particular OCAM developed by [SAKA70], and this in turn to management of indexes for data/knowledge bases.

II. HOLOGRAPHY

Introduction

In order to establish why holography is useful in storing information we draw an analogy between holography and photography. A photograph captures a light image, and when it is developed we can see that image. Photography basically provides a method of recording the 2-dimensional irradiance distribution of the image. Generally speaking

each "scene" consists of a large number of reflecting or radiating points of light. The waves from each of these elementary points all contribute to a complex wave, which is called the "object wave". This complex wave is transformed by the optical lens in such a way that it collapses into an image of the radiating object. It is this image that is recorded on the film.

Holography is quite different. A hologram [GABO69] records the whole of the wave information about the light wave at each point on a 2-dimensional surface, in other words the object wave itself is recorded and not the optically formed image of the object. This wave is recorded in such a way that a subsequent illumination of this record serves to reconstruct the original object wave. A visual observation of this reconstructed wavefront then yields a 3 dimensional view of the object or scene, which is practically indiscernible from the original. It is thus the recording of the object wave itself, rather than an image of the object, which constitutes the basic difference between conventional photography and holography.

Basics of Recording and Reading a Hologram

The basic technique of forming a hologram is shown in Figure 37. The coherent light beam coming from a laser is divided into two beams. One of the beams is used to illuminate the object and the other acts as a reference, and therefore are referred to as the *object beam* and *reference beam*, respectively.

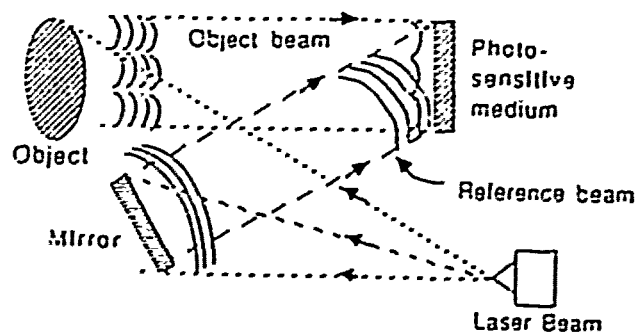


Fig-37. Formation of a Hologram

The reference beam is directed so that it will intersect and overlap the object beam at a point where a photosensitive medium has been placed. The reference beam and the object beam, will form an interference pattern on the photosensitive medium. After the beams are removed the medium is processed and the interference pattern formed by the overlapping of the object and reference beams is called a *hologram* [COLL79].

A hologram is read by illuminating it with a beam having the same physical properties as the reference beam with which it was recorded. As the beam passes through the hologram it is diffracted into a recreation of the original object wave coming from the object.

An observer viewing a wave identical to the original object wave, perceives it to diverge from a virtual image of the object located precisely at the original object's location. On the other hand if the hologram's backside is illuminated by a conjugate reference beam (a beam in which all the rays are exactly opposite to the original reference beam), then a real image is formed at the original object's position, as shown in Figure 38. Since the light converges to a real image, the image can be detected by photodetectors. Hence, a hologram is a diffracting record of the interference of an object and reference beam.

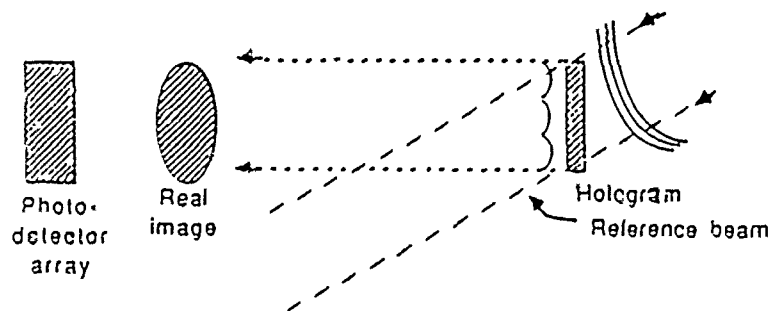


Fig 38 Reading a Hologram

Page Holograms: 2-Dimensional Recording

A *page hologram* can be defined as a hologram made up of many nonoverlapping subholograms or pages [RAJC70]. A page is recorded through the use of a page com

poser, which is a device that stores data and converts it from electronic form to optical form, as indicated in Figure 39.

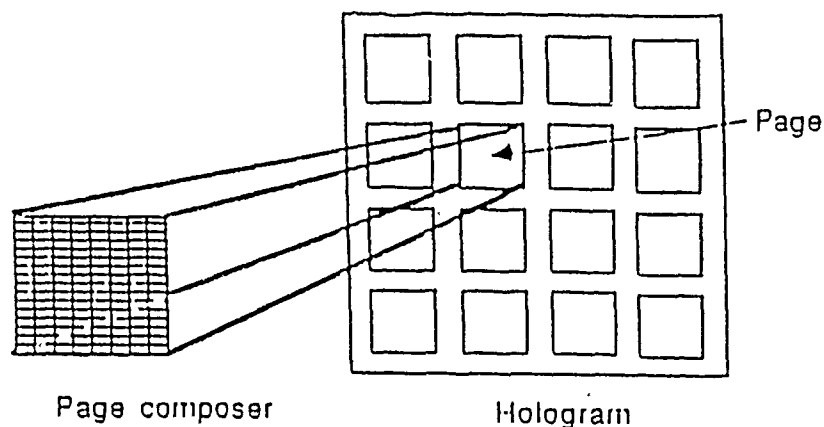


Fig 39. Page Hologram with Page Composer

To record a page, data is first loaded from the computer to the page composer, then a small area of the emulsion is exposed by an aperture, and the information stored in the page composer is flashed onto the exposed area. To record another page the aperture is moved, new data is loaded into the page composer and the process is repeated. Depending on the medium used, information can be erased from a page, and the page can be rewritten with new data.

Arrays of page oriented holograms can hold large amounts of data in a small area. For example typical values for a page are one millimeter on a side with a capacities of around 10^4 bytes. Assuming that hologram arrays can hold pages on two millimeter centers, a 3 X 5 inch card would hold 2400 pages, for example data on the card would be about 24 Megabytes. We should point out that these page oriented holograms are not readily available in the commercial market but in the laboratory.

III. Storage and Readout Based on the Exclusive OR Principle

In this section we explain how data are represented and then recorded in the hologram. Let us say that we have data to record, call it A. The A information is represented by beam $A(x)$, which consists of $2n$ beams, $\bar{a}_1, a_1, \bar{a}_2, a_2, \bar{a}_3, a_3, \dots, \bar{a}_n, a_n$. Each bit of A is

represented by a true beam a_i and a complement beam \bar{a}_i , where i represents the i th bit. The light beams that make up $A(x)$ come from the true or complement bit places, according to whether the corresponding bits are in states "1" or "0". Suppose that (A_1, A_2, A_3, A_4) is (1,0,1,0) then, $A(x)$ is a set of beams coming from $a_1, \bar{a}_2, a_3, \bar{a}_4$ that can be used to record the A data, as shown in Figure 40.

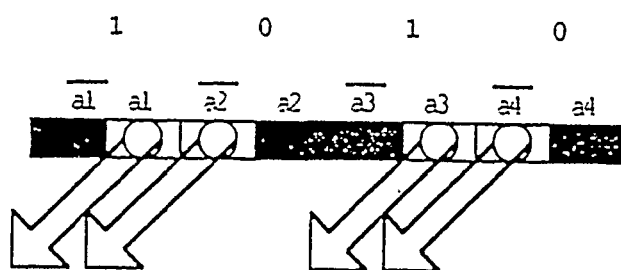


Fig 40. Data Representation

The holograms are read by interrogating them with a search argument $C=(C_1, C_2, C_3, \dots, C_n)$, represented by beam $C(x)$. Beam $C(x)$ consists of $2n$ light beams, $\bar{c}_1, c_1, \bar{c}_2, c_2, \bar{c}_3, c_3, \dots, \bar{c}_n, c_n$. A match is detected by the exclusive Or principle; that is, a match occurs if the i th interrogation signal bit and the i th interrogated bit are different. So, in actuality the beam that does the interrogation, $C'(x)$, is the complement of beam $C(x)$. The corresponding c_i or \bar{c}_i beams simultaneously illuminate the corresponding a_i or \bar{a}_i bits of all the words subjected to interrogation. Some of the C_i 's can be DON'T CARES (dc); as such they do not participate in the interrogation so no light is emitted from the corresponding c_i or \bar{c}_i bit. To illustrate the point the previous value of A is used, where $A = (1,0,1,0)$. If the search argument (C_1, C_2, C_3, C_4) is $(1,0,dc,0)$, then the corresponding interrogation beam $C'(x)$ is actually $(0,1,dc,1)$; that is, it consists of three parallel beams coming out of \bar{c}_1, c_2 and c_4 positions, with no light coming from c_3 or \bar{c}_3 , because the third bit is a don't care. We find that the interrogation signal $C = (1,0,dc,0)$ matches the interrogated signal $A = (1,0,1,0)$, in reality it is the EXOR between $C' = (0,1,dc,1)$ and $A = (1,0,1,0)$, which determines whether a match has occurred or not. The match is detected by a photodetector array; wherever the detector detects a null then a match has occurred. If all the bits for a word are detected as null

then, the word matches the search argument.

EXOR Table Look-Up

In this section we describe an associative system, which is based on table look-up. In the table look-up procedure, given a particular search argument we compare it against reference patterns stored in a hologram. Detection of a match is done using the EXOR principle.

The table look-up method has two components. The first component is the memory composed of holograms, and the second is the page composer. The holograms contain the reference patterns that the search argument will be compared against. A different reference pattern is recorded in each row of the hologram. From now on we will refer to the hologram as the table.

The second component, the page composer, contains the search pattern, which will be compared against the reference patterns stored in the hologram. The search argument is replicated in all the rows of the page composer. When the search is performed each row of the page composer is compared with each row of the table. If a null row is detected by the photo-detector array, then a match has occurred.

To illustrate this let us take a simplified example (Figure 41). For illustration we ignore the actual coding of bits. The table contains the names of the people working in a department. Each name corresponds to a reference pattern. Only the names of people actually working in the department are recorded as reference patterns. Suppose that now we want to find out whether or not a person of a given name, John, is working in the department. John is loaded into the page composer as shown below. The search argument is flashed against the table. If there is a person named John in the department then a null will occur and it will be detected by the photo-detector array. In the example a null is detected in the third row. Since a null row has been detected, the answer is yes; someone with the name of John works in the department.

john
john
john
john
john

page composer

cathy
tom
john
louis
susan

holographic
table

not null
not null
null
not null
not null

photo-detector
array

Fig 41. EXOR Table Look-Up Example

IV. Associative Memory

In this section we discuss how optics can be used in implementing the various associative memory operations associated with conventional CAMs. Associative operations can be grouped into 4 categories as follows: Equivalence Searches, Threshold Searches, Double Limit Searches, and Extremum Searches. Algorithms to perform these operations in electronic CAMs have been proposed by [RAMA78] and [DAVI86].

Equivalence Searches

We now demonstrate how equivalent searches can be implemented using OCAMs and the table look-up method. The lower part of Figure 42 represents the words stored in our associative memory or table. For illustrative purposes we limit the number of words to seven of which each word is five bits in length ($n=5$). The light and dark patterns follow Figure 40 and bit values (1 and 0) are included for convenience. Our search argument (S) is 10110 as shown in the upper portion of Figure 42. We can mask (M) any bit as can be done in electronic CAMs by just using DON'T CARES. If we mask bit three, the resultant search argument (S') is 10dc10. Recalling from the EXOR section the actual search argument (SEXOR), is the complemented value as shown in Figure 42. This complemented value is loaded into all the rows (seven in our example) of the page composer.

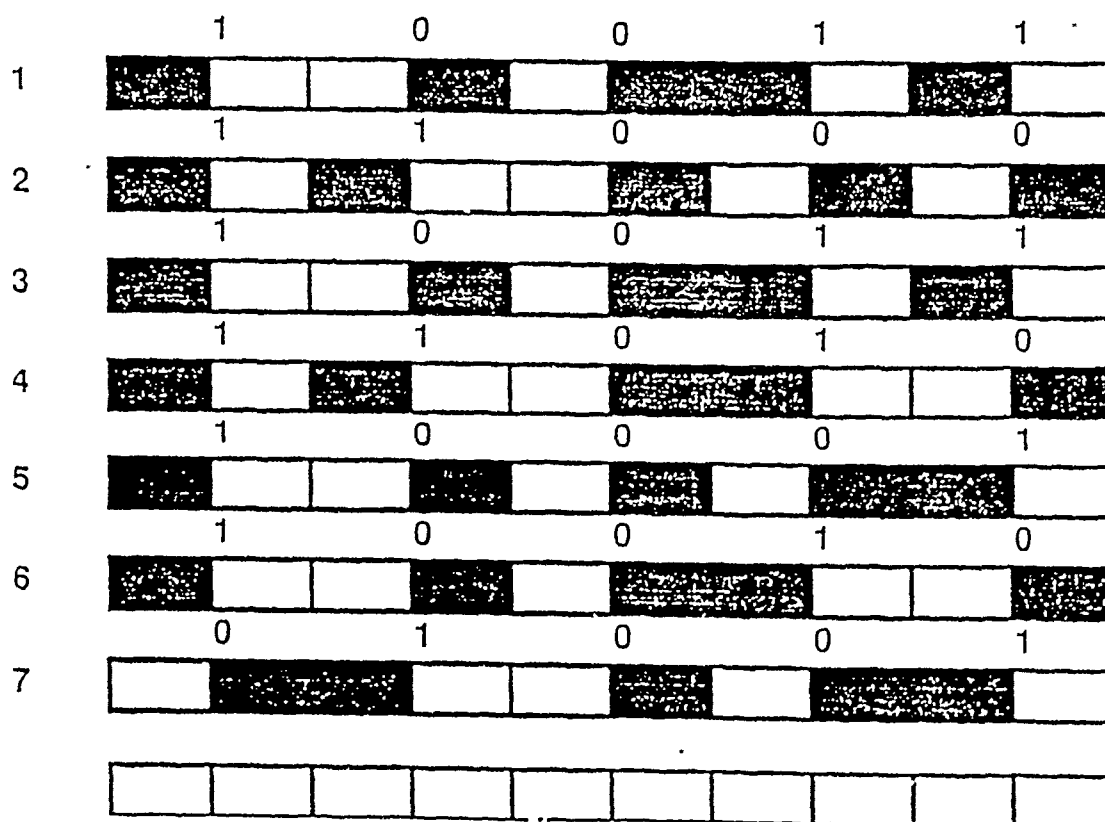
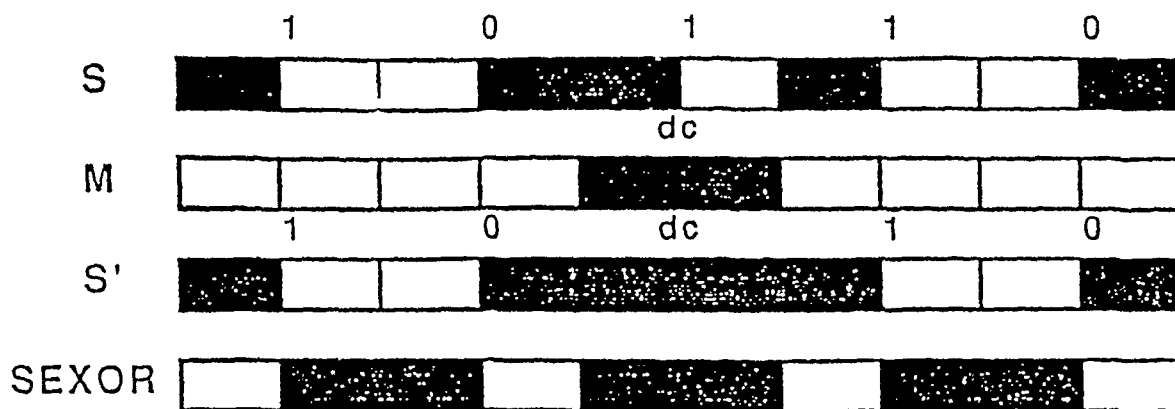


Fig 42. Search Arguments and Associative Memory or Table

The next step is to flash the page composer contents (seven copies of the search argument) against the table. An EXOR is performed between the corresponding bits in the page composer and the table. Figure 43 shows the effect of this operation, as the photo-detector array would see it.

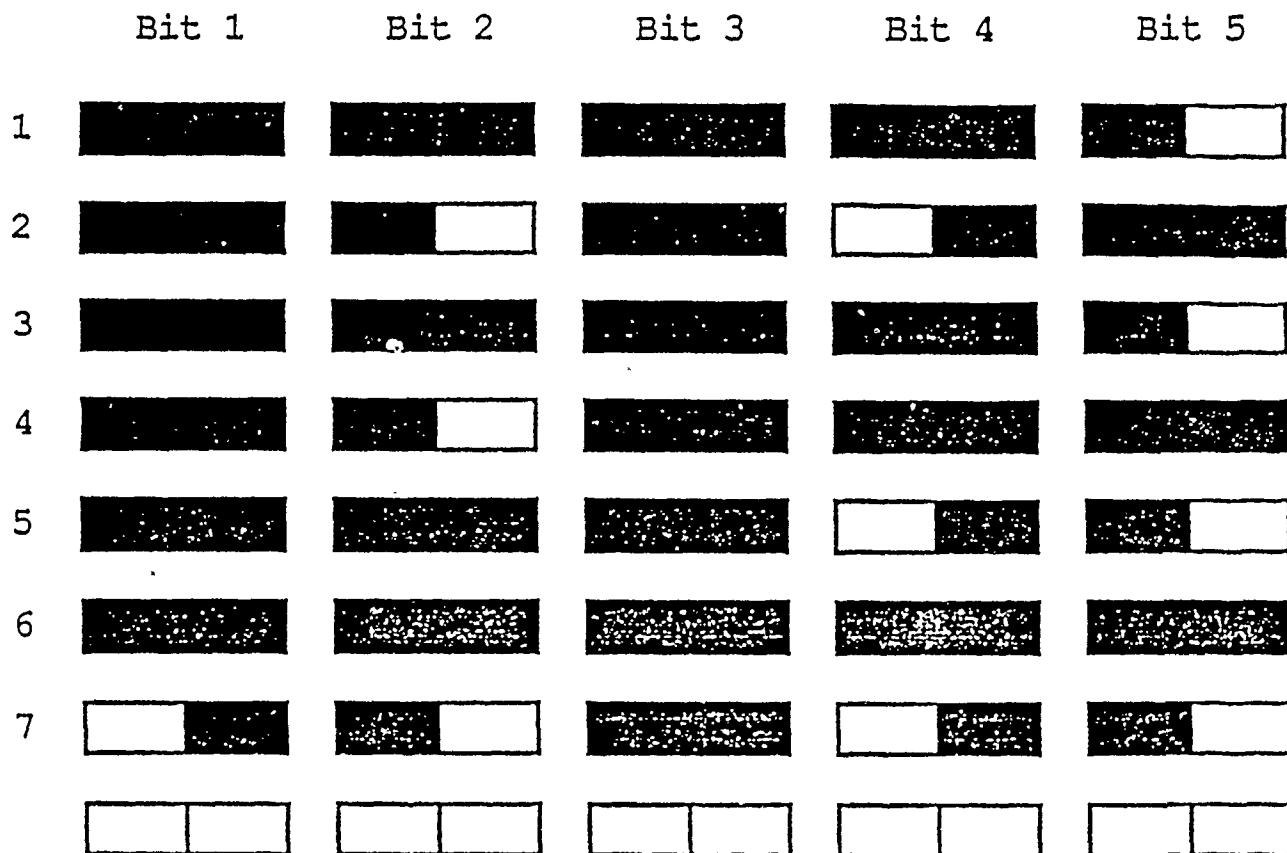


Fig 43. Results of EXOR Table Look-Up at Photo-Detector Array

As mentioned previously any row containing all nulls is a match. We see that row 6 is a null row, and therefore is equal to the search argument S' . The inequality match is just the opposite interpretation of the equality criteria.

Threshold and Double Limit Searches

In this section we show how to perform some of the more complex associative operations. We begin by showing how the words in memory (table) can be divided according to whether they are Less-Than, Equal-To, or Greater-Than the search argument, denoted by L, E, and G respectively. From now on we will refer to this as LEG. Our OCAM approach is based on the LEG concept. Once the search space is divided into LEG the other more complex operations fall right out, since these operations are composed of the unions and intersections of repeated LEG operations.

To divide the search space (SS) into LEG, we will operate in bit serial fashion. The search argument is loaded into the page composer and replicated as before, but instead of flashing all the search argument bits against the table, we start with the most significant bit and operate in bit serial fashion down to the least significant bit. We flash the *i*th bit column of the page composer against the *i*th bit column of the table. The resulting beams are the same as those in Figure 43, but taken a bit column at a time. For convenience we use the notation shown in Figure 44.


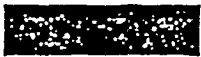

		L	E	G
The bit is less than the search argument bit.		1	0	0
The bit is equal to the search argument bit.		0	1	0
The bit is greater than the search argument bit.		0	0	1

Fig 44. LEG Coding Scheme

The steps to divide the search space into LEG can be summarized by the algorithm below.

Algorithm

n bit word

load the search argument into the page composer and replicate

for i=1 to n do

begin

- flash the ith bit column contents of the page composer against the ith table bit column
- determine any matches between the search and table bits
for any row where a null is *not* detected by the photo-detector array do

begin

- interpret the beams striking that photo-detector array row according to LEG coding, and update the LEG Word.

- disable that particular photo-detector array row.

end

end

collect responses to group the words according to LEG.

We can illustrate this algorithm with an example as indicated in Figure 45. The search argument is loaded into the page composer and replicated. Starting with the most significant bit, the first bit column, we flash it against the most significant bit column of the table. A photo-detector array detects any matches (null rows) and non matches (non-null rows). If a match has occurred no judgement can be made whether the word is LEG than the search argument. However, if a non-null row appears then we can interpret the beams striking the photo-detector array according to the LEG coding scheme (Figure 44). From this information we can determine whether that word is Less-Than or Greater-Than the search argument. As we see in Figure 45 a non-null in the seventh row is detected by the photo-detector array. This non-null is interpreted by the detector array and Word LEG is updated according to the LEG coding scheme as being Less-Than the search argument. The seventh row of the photo-detector array is then disabled, because this non-null contained the information to classify the word

Photo-detector
array

Word LEG

0 1 0
0 1 0
0 1 0
0 1 0
0 1 0
0 1 0
1 0 0

not null

0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
1 0 0

0 1 0
0 0 1
0 1 0
0 0 1
0 1 0
0 1 0

not null

not null

0 0 0
0 0 1
0 0 0
0 0 1
0 0 0
0 0 0
1 0 0

0 1 0
0 1 0
0 1 0
0 1 0

0 0 0
0 0 1
0 0 0
0 0 1
0 0 0
0 0 0
1 0 0

0 1 0
0 1 0
1 0 0
0 1 0

not null

0 0 0
0 0 1
0 0 0
0 0 1
1 0 0
0 0 0
1 0 0

0 0 1
0 0 1
0 1 0

not null

not null

0 0 1
0 0 1
0 0 1
0 0 1
1 0 0
0 0 0
1 0 0

Word LEG

FINAL RESULT

0 0 1
0 0 1
0 0 1
0 0 1
1 0 0
0 1 0
1 0 0

Fig 45. LEG Example

in row seven as being Less-Than or Greater-Than the search argument.

The second bit column is flashed against the table, but since the seventh row of the photo-detector has been disabled, only the remaining six rows will be detected. We see that there are non-nulls in rows two and four. The beams striking photo-detector array rows two and four are interpreted and the corresponding rows in Word LEG are updated according to the LEG coding scheme. Next the third bit column is flashed, only rows 1, 3, 5, and 6 are detected. Only nulls occur at these rows, as it should be since this bit was a DON'T CARE. We proceed in this fashion until we have looked at all the bits, as seen in Figure 45. The Word LEG is now complete.

At this point we have divided the memory into three sets: Less-Than, Equal-To, or Greater-Than the search argument, L, E, G, respectively. Based on combinations of these sets we can implement the other more complex associative searches, as is summarized in Figure 46. For the Double-Limit Searches, where the limits are denoted by X and Y, two searches must be performed, one for the lower limit, and another search for the upper limit. These searches can also be done in parallel, if provisions are made to include an extra set of page composers and tables. For these searches we have two sets of results, the first set is for the limit X denoted by, LX, EX, and GX. The second set is for the limit Y denoted by, LY, EY, GY.

Extremum Searches

In this section we present an algorithm to perform the Extremum searches, maximum and minimum, using OCAMs and the table look-up method. To demonstrate the search we will use the words in memory as shown in Figure 42. Let us first discuss the Maximum search. For this search we use as the search argument, a word consisting of all ones and proceeding in bit serial fashion. We record the maximum value bit by bit in the K register, which is of length n, according to the following algorithm.

I Equivalence Searches

1. Equality Search: detection of null rows
2. Inequality Search: detection of non-null rows

II. Threshold Searches

1. Greater-Than Search: G
2. Less-Than Search: L
3. Greater-Than-Or-Equal-To Search: $G \cup E$
4. Less-Than-Or-Equal-To Search: $L \cup E$

III Double Limit Searches

S' is the Search Argument

A. Between-Limits Search, $X < Y$

1. $S' > X$ and $S' < Y$: $LY \cap GX$
2. $S' > X$ and $S' \leq Y$: $GX \cap (EY \cup LY)$
3. $S' \geq X$ and $S' < Y$: $(GX \cup EX) \cap LY$
4. $S' \geq X$ and $S' \leq Y$: $(GX \cup EX) \cap (LY \cup EY)$

B. Outside- Limits Search, $X < Y$

1. $S' < X$ or $S' > Y$: $LX \cup GY$
2. $S' < X$ or $S' \geq Y$: $LX \cup EY \cup GY$
3. $S' \leq X$ or $S' > Y$: $LX \cup EX \cup GY$
4. $S' \leq X$ or $S' \geq Y$: $LX \cup EX \cup EY \cup GY$

Fig 46. Summary of Complex Searches

Algorithm

n is the bit length of the word

K register of length n to record the maximum word

load all 1's as the search argument into the page composer

for i=1 to n do

begin

- flash the contents of the ith bit column of the page composer against the ith bit table column

- determine any matches between the ith search and table bits
if a null has occurred at any bit row do

begin

- set ith bit in the K register to one.

- disable all the rows of the photo-detector array where a no match (non-null) has been detected (the rest of the rows are still candidates).

end

if no matches (non-nulls) have occurred do

begin

- set the ith bit in the K register to zero.

- do not disable any rows (all rows are still candidates).

end

end

K register contains the maximum word in the search space

Figure 47 shows the determination of the maxima for the example given in Figure 42.

Bit 1		Bit 2		Bit 3		Bit 4		Bit 5
0 1 0	null	1 0 0						
0 1 0	null	0 1 0	null	1 0 0		1 0 0		
0 1 0	null	1 0 0						
0 1 0	null	0 1 0	null	1 0 0		0 1 0	null	1 0 0
0 1 0	null	1 0 0						
0 1 0	null	1 0 0						
1 0 0								
K								
1		1 1		1 1 0		1 1 0 1		1 1 0 1 0

Fig 47. Maximum Search Example

The algorithm for Minimum search is essentially the same as the Maximum search, except the search argument now consists of all zeroes instead of all ones. When a match does occur for the i th bit, then the i th bit of the K register is set to 0, and is set to one if no matches occur. Once again, at the completion of the algorithm, register K contains the minimum word in the memory space.

Depending on how the associative memory is organized i.e. fields, then we will have to load the maximum (minimum) word into the page composer, and determine its location. Having determined the location we can readout the other fields, which may have been masked out.

V. DATABASE INDEX USING OCAMs

In the previous section we have shown how to perform various associative searches on Optical Content Addressable Memories (OCAMs) using a table look-up technique. In this section we present a holographic technique that can be used for managing an index to a very large database. [SAKA70] demonstrated a relatively straight forward extension of conventional holographic techniques in order to develop a one dimensional optical content addressable memory. Later [KNIG74] refined the idea into a

page oriented holographic associative memory. [BERR88] showed how Sakaguchi's and Knight's associative memory could be applied to managing an index to a database.

Sakaguchi used two object beams $A(x)$, $B(x)$ and one reference beam, instead of one object and one reference beam to record a thin hologram. Information A, is represented by beam $A(x)$, which consists of $2n$ (n is the number of bits) coherent parallel beams, while the B information is represented by beam $B(x)$ consisting of $2m$ (m is the number of bits) diverging beams. The information in beams $A(x)$, and $B(x)$ is represented in binary form as described in the previous section.

Knight (1974) extended Sakaguchi's work by recording a page hologram consisting of 100 by 100 pages each containing 10^4 bits/page, for a total of 10^6 bits. Each page was very small; 1mm^2 . In recording the 10,000 pages the $A(x,y)$ and $B(x,y)$ beams were kept stationary while the reference beam was repositioned for each page. The page composer was loaded with new $A(x,y)$ and $B(x,y)$ data for each page.

What makes this suitable for implementing a database index is that each page has a code word, A, associated with it; and the data, B, written in a particular page can be identified by using the code word. In other words, the entire memory plane can be searched with a code word represented by beam $A(x,y)$ in parallel with one flash of light. When a match is detected by the photo-array, the data, represented by beam $B(x,y)$, corresponding to that page is read out. Next, an example will be presented that illustrates how this technique can be applied in implementing an index scheme.

In this example we use an inverted list with indirect addressing as the index technique. This particular index was chosen over the more conventional inverted list index, due to the fact that whenever a reorganization of the database occurs (the physical addresses of the records change), the addresses in the holograms do not have to be changed. This is of particular importance since the hologram is read only.

Database						
SS#	Name	Sex	Job	Dept#	Salary	Seniority
012345678	Mary	F	SQ	120	1000	73
123456789	Gary	M	FI	110	2000	64
345678901	John	M	EN	110	1750	76
456789010	Peter	M	JN	130	900	84
567890123	Eva	F	FI	120	2400	79
678901234	Teresa	F	EN	120	1750	81
789012345	Sophie	F	JN	120	1000	83
890123456	Mark	M	SQ	130	1200	77
901234567	Laura	F	FI	110	2200	76
946801234	Hector	M	EN	130	2000	72
956801234	Bruce	M	EN	110	3000	70
968012345	Luke	M	FI	120	2400	74
979123456	Dean	M	SQ	130	1300	74
980123456	Sue	F	EN	120	2000	82
991245678	Cindy	F	JN	130	1100	85
999234567	Anna	F	JN	120	1200	83
999379135	Sam	M	EN	130	2300	80
999387913	Lucy	F	FI	110	2200	80
999581470	Ken	M	SQ	130	1300	72
999826048	Henry	M	FI	110	2400	70

Fig 48. Database Excerpt

The database shown in Figure 48 is an employee database, and indexes are formed on SS#, Sex, Job, Dept# as shown in Figure 49. We now apply Knight's work to the above database index scheme. The pages in the hologram represent the attributes we have indexed on. For example looking at the Department index we see that there are 3 different departments (110, 120, 130). With each department number we have associated a list of serial#'s. In recording the data we place the department number on the A(x,y) beam and the list of serial#'s associated with that particular department number on the B(x,y) beam. The representation is stored as shown in Figure 50.

Associative Searches

Associative operations can be performed on the keywords, so as to find the page that contains the desired information. For example, suppose we need to find some information about the persons working in Dept 110. To find the page or pages that contain the serial#'s an equivalence search has to be performed between '110' and all the keywords. The interrogation signal is flashed onto the memory plane, and the resultant

A1 Index			A2 Index		A3 Index		A4 Index	
Serial#	SS#	Address	Sex	Serial#	Job	Serial#	Dept#	Serial#
1	012345678	32	M	2	SQ	1	110	2
2	123456789	23		3		8		3
3	345678901	14		4		13		9
4	456789010	43		8		19		11
5	567890123	16		10	FI	2		18
6	678901234	10		11		5		20
7	789012345	6		12		9	120	1
8	890123456	19		13		12		5
9	901234567	22		17		18		6
10	946801234	13		19		20		7
11	956801234	45	F	20	EN	3		12
12	968012345	4		1		6	130	14
13	979123456	12		5		10		16
14	980123456	21		6		11		4
15	991245678	30		7		14		8
16	999234567	36		9	JN	17		10
17	999379135	40		14		4		13
18	999387913	25		15		7		15
19	999581470	18		16		15		17
20	999826048	7		18		16		19

Fig 49. Indexes

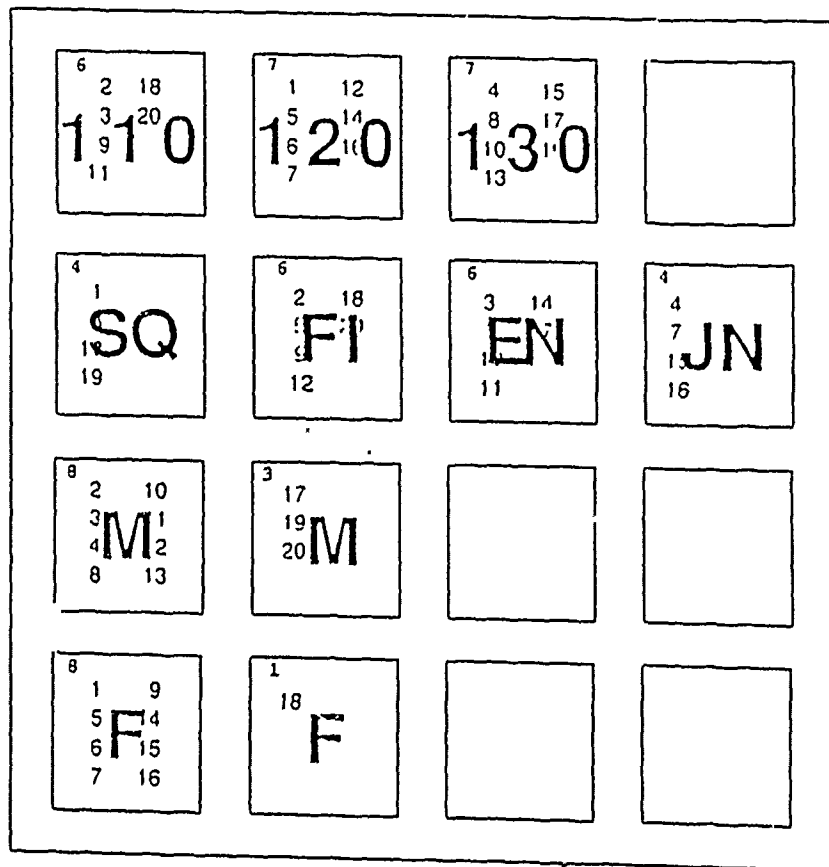


Fig 50. OCAM Representation

beams are detected by a photo-detector array. If a null word is detected, then the page corresponding to this keyword position contains the information desired. Having located the page, it is now illuminated and the information is read out in parallel and at the speed of light. The data concerning department 110 may span several pages. In that case each of the pages must be read in turn to get the serial#'s. The serial#'s are now used to obtain the data records. Obviously, a wide variety of query types can be executed and a more complex example is given in the next section.

Conjunctive Query

In this section we illustrate how a conjunctive query can be performed using the techniques and algorithms described in the previous sections. We use the following procedure

1. Flash in sequence all the search arguments (keys) needed to do the complex query.
 - For each key value note the position and the number of page matches.
2. Order the keys by the number of pages of information they have, least to most.
 - If any have the same number of pages, read the reserved word to determine the amount of data on the page.
3. Read into electronic memory the pages for the key with the least amount of data. This is the source of search arguments for the page composer; the others will be the table(s).
4. Do until all the information from the search argument page is exhausted
begin
 - load the search argument values one by one into the page composer.
 - compare the values against the tables (least to most amount of pages).
 - if a match is detected, then proceed to do comparisons with the next table, else no need to go further.end
5. If a search argument matches data on all the pages it satisfies the conjunctive query.

To illustrate the above procedure, we perform the following query:

Find all the people who work in Dept 120, who are female (F),
and whose job is finance (FI).

We would in sequence flash these search arguments (keys) against the memory plane such as the one in Figure 50. We would note the position and number of the matches. In our example, for the keys 120, F, and FI, we have 1, 2, and 1 page(s) containing the respective serial# information. We will use the keys which have the largest number of pages as the tables, and the one with the least as the search argument.

The tables will be searched in the order from least to most tables; definitely F will be used as a table, but whether to use FI or 120 as a table could be an arbitrary choice. However, the most time consuming operation is the loading of the page composer with the search argument. For example, if FI had only one entry and 120 had one hundred entries in their respective pages then clearly, it would be desirable to use the information associated with 120 as the table, and the information associated with FI as the source for the search arguments. By choosing 120 as the table we need to only do one page composer load, and a parallel search of all one hundred words is performed, versus one hundred page composer loads and searching only one word at a time if FI was chosen as the table. Therefore, it is beneficial to have the first word in each page be a reserved word, containing the number of serial#'s recorded in that page. In our example FI contains six, while 120 contains seven serial#'s. So the information associated with key FI is read into electronic memory to be used as the source for the search arguments.

Figure 51 illustrates the setup used. To find the answer to the query we proceed as follows. Having read out the information into electronic memory for FI, we take the first serial# and load it into the page composer bank where it is replicated. The number of page composers used in the bank depends on how many pages each table is composed of; one page composer for each page in a table. In our case to do a table look-up against 120 one page composer needs to be loaded with the search argument. The

argument is flashed against the table. In this action we are checking whether the person whose job is FI also works in department 120. If a match does occur, then we proceed to do another table look-up to check whether this same person is female (F). However, if no match had occurred in the 120 table look-up, there would have been no need to do the F table look-up, then the page composer bank would have been reloaded with the next person working in Dept 120. The bank of Bragg Cells steer the search argument light beams coming from the page composer to the proper table. In our example the first serial# from FI to be loaded into the page composer is a '2'. Doing the comparison against table 120, we find that no match occurs, therefore the page composer is loaded with the next search argument, serial# 5. Performing a comparison with serial# 5 against the 120 table, we find that a match occurs. Since a match has occurred we proceed to do a comparison with serial# 5 against the F table. Doing the comparison we find that we have a match, and therefore serial# 5 satisfies our conjunctive query.

The above process could obviously be speeded up by pipelining the table look-ups. Such a speed-up would depend on how many page composer banks the system would be provided with. One set of page composer banks can be doing table look-ups while the banks are being loaded.

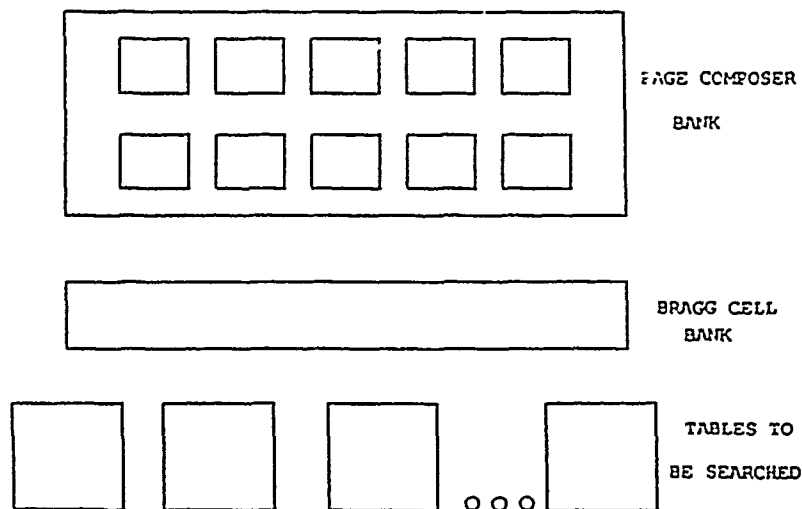


Fig 51. Conjunctive Query Setup

VI. CONCLUSION

In this paper we have introduced the notion of how OCAMs can be used in managing an index to a very large data/knowledge base. Obviously, if the data/knowledge base can fit into the OCAM other approaches can be taken. OCAMs have many attributes, large size, almost instantaneous access and parallel readout, that are important to solving the problems associated with managing an index for very large data/knowledge bases.

The equivalence, threshold, double limit, and extremum searches are well suited for CAM processing. However, due to the expensive nature of electronic CAMs, only a small CAM can be included in any system. The problem with a small CAM for data/knowledge base applications is that it is input output bound. Due to the above detractions the use of electronic CAMs for data/knowledge base applications has not met with a great deal of success.

We present OCAMs as an alternative to electronic CAMs for data/knowledge base applications. OCAMs have the potential for holding megabytes of data at an appreciably lower cost than electronic CAMs. OCAMs perform optically all the various operations associated with conventional CAMs, and in addition they provide parallel output. Holograms do however suffer from the fact that it takes a long time to form them and they are difficult to change once formed. However, for very large data/knowledge bases indexing structures can be devised that are rather insensitive to updates provided that the update rate remains moderate. We are continuing our research into how OCAMs can be applied to other data/knowledge base operations.

REFERENCES

- [BERR88] P.B. Berra and S.J. Marcinkowski, "Optical Content Addressable Memories for Managing an Index to a Very Large Data/Knowledge Base", Data Engineering Bulletin, March 1988 Vol.11 No.1
- [COLL71] R.J. Collier, C.B. Burckhardt, L.H. Lin, *Optical Holography*, Academic Press Inc., 1971
- [DAVI86] Wayne A. Davis and De-Lei Lee, "Fast Search Algorithms for Associative Memories", IEEE Transactions on Computers, Vol. C-35 No.5 May 1986
- [GABO69] D. Gabor, "Associative Holographic Memories", IBM Research Journal, March 1969
- [KNIG74] Gordon R. Knight, "Page-Oriented Associative Holographic Memory", Applied Optics, Vol. 13, No. 4, April 1974
- [RAJC70] J.A. Rajchman, "An Optical Read-Write Mass Memory", Applied Optics, Vol. 9, No. 10, October 1970
- [RAMA78] C.V. Ramamoorthy, James L. Turner, Benjamin W. Wah, "A Design of a Fast Cellular Associative Memory for Ordered Retrieval", IEEE Transactions on Computers, Vol. C-27, No. 9 Sept. 1978
- [SAKA70] M. Sakaguchi, N. Nishida, T. Nemoto, "A New Associative Memory System Utilizing Holography", IEEE Transactions on Computers, Vol. C-19, No. 12, December 1970

Optical Content Addressable Memories for Managing an Index to a Very Large Data/Knowledge Base

P. Bruce Berra and Slawomir J. Marcinkowski

Dep't of Electrical and Computer Engineering
Syracuse University
Syracuse, NY 13244-1240
U. S. A.

ABSTRACT

In this paper we present an optical approach for managing an index to a very large data/knowledge base. Specifically, our research is focused on optical content addressable memories (OCAM) based on holographic principles. OCAMs offer advantages over conventional CAMs in managing a large index, because OCAMs offer a considerably larger storage capacity (10^8 - 10^{10} bytes) and parallel output. With the large capacity of the OCAM, the potential exists for storing full index data to a very large data/knowledge base, which would otherwise be prohibitively costly in conventional electronic CAMs.

INTRODUCTION

One of the major problems in the management of very large data and knowledge bases (10^{12} - 10^{16} Bytes) is the time needed to access the desired data/knowledge. There exist memory hierarchies along with various indexing and search schemes which are designed to minimize the time it takes to access the desired data/knowledge. These include: hashing, B-trees, inverted lists, pointer systems, etc. All of these techniques have their strengths and weaknesses; so no one method is optimal for all situations. However, due to the fact that there is an increase in performance when using an index over a full sequential search of the entire data/knowledge base all data/knowledge base management systems have the facility to construct an index. While retrieval performance can be improved through the use of indexing it is not without its costs. The main costs are associated with the update of the index data when changes, additions, and deletions are required and in the management of the index data due to its large physical size. In fact, in some applications where considerable index data are required, the size of the indexes will be as large as or even larger than the data/knowledge being managed.

Research over the years has focused on how to improve the access time without incurring too much overhead in terms of update and storage space required for indexing data. Content-addressable memories (CAM), a hardware technique, have attracted a lot of attention. The CAM is attractive to database applications, because data can be accessed based on content, and not by addressing. Accessing data by content offers considerable speed up, because the search is conducted in

parallel. Also, various operations can be performed in searching the data depending on the application. For instance, in certain computations such as text retrieval it is only necessary to locate entries which exactly match a given search argument. In other database applications, the equality search is not the only type of search performed. Frequently, there occurs a need to do a masked search involving only a subset of the search argument, or to locate entries which satisfy specified magnitude relations, for instance, having one or several of their attributes above, below, or between specified limits, or absolutely greatest or smallest. The CAM is well suited to the performance of these types of operations. In fact, CAMs can be used to perform match (equality), mismatch (not equal to), less than, less than or equal to, greater than, greater than or equal to, maximum, minimum, between limits, outside of limits, next higher, next lower, and various forms of add/subtract.

However, CAMs are not without their disadvantages. The chief constraint on CAMs is that they are quite expensive, and therefore only a small CAM can be included in any system. The problem of a small CAM is that for data/knowledge base applications the CAM is input output bound. The performance of the CAM is governed by the time it takes to load the data into the CAM, rather than the time to process CAM resident data. In fact, this disparity may be several orders of magnitude. It is for this reason that CAMs have not found widespread use in data/knowledge base systems.

The above comments apply to CAMs that are constructed from electronic components. However, CAMs can be constructed from optical devices, particularly holograms, and these are called optical content addressable memories (OCAM). They have the desirable search processing properties of electronic CAMs yet offer a considerably larger storage capacity (10^8 - 10^{10} bytes) and parallel output. Their main disadvantage is that they are read only. However, if the database under consideration is static and requires fast access then OCAMs offer a potentially cost effective solution.

In our research, we are considering OCAMs for the processing of index data to a very large data/knowledge base. With the large capacity of the OCAM, the potential exists for storing full index data to a very large data/knowledge base. We are taking a page oriented approach which serves to insulate us somewhat from changes.

In this paper we provide an overview of holography and some of its characteristics. We then discuss a particular OCAM developed by Sakaguchi 1970. We show how it might be used to implement an indexing scheme and discuss some of the issues that one must consider in managing the index to a very large data/knowledge base with OCAMs.

HOLOGRAPHY

Introduction

The term holography is derived from the Greek "holos" meaning "the whole or entirety" [Gabor 69]. A hologram records the whole of the wave information about the light wave at each point on a 2-dimensional surface. The wavefront can easily be reconstructed so that the 3-dimensional information about the wave can be retrieved at will.

In order to establish why holography is useful in storing information we draw an analogy between holography and photography. A photograph captures a light

image, and when it is developed we can see that image. Photography basically provides a method of recording the 2-dimensional irradiance distribution of an image. Generally speaking each "scene" consists of a large number of reflecting or radiating points of light. The waves from each of these elementary points all contribute to a complex wave, which is called the "object wave". This complex wave is transformed by the optical lens in such a way that it collapses into an image of the radiating object. It is this image that is recorded on the photographic emulsion (film).

Holography is quite different. With holography, one records the object wave itself and not the optically formed image of the object. This wave is recorded in such a way that a subsequent illumination of this record serves to reconstruct the original object wave. A visual observation of this reconstructed wavefront then yields a view of the object or scene which is practically indiscernible from the original. It is thus the recording of the object wave itself, rather than an image of the object, which constitutes the basic difference between conventional photography and holography sometimes referred to as "lensless photography" [Smith 69],[Caulfield 70].

Basics of Recording a Hologram

The basic technique of forming a hologram is shown in Figure 52. The coherent light beam coming from a laser is divided into two beams. One of the beams is used to illuminate the object and the other acts as a reference, and therefore are referred to as the *object beam* and *reference beam*, respectively.

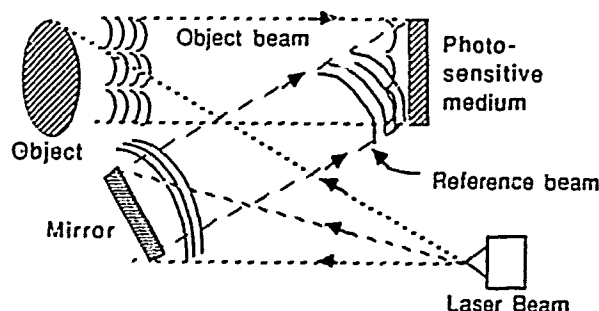


Fig 52. Formation of a Hologram

The reference beam is directed so that it will intersect and overlap the object beam at a point where a photosensitive medium has been placed. The reference beam and the object beam, will form an interference pattern on the photosensitive medium. After the beams are removed the medium is processed and the interference pattern formed by the overlapping of the object and reference beams is called a *hologram* [Collier 79].

Basics of Reading a Hologram

A hologram is read by illuminating it with a beam having the same physical properties as the reference beam with which it was recorded. As the beam passes through the hologram it is diffracted into a recreation of the original object wave coming from the object.

An observer viewing a wave identical to the original object wave, perceives it to diverge from a virtual image of the object located precisely at the original object's

location. On the other hand if the hologram's backside is illuminated by a conjugate reference beam (a beam in which all the rays are exactly opposite to the original reference beam), then a real image is formed at the original object's position, as shown in Figure 53. Since the light converges to a real image, the image can be detected without the use of a lens by photodetectors. Hence, a hologram is a diffracting record of the interference of an object and reference beam.

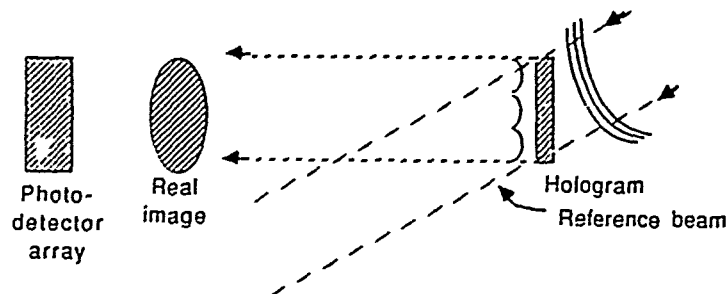


Fig 53. Reading a Hologram

Page Holograms: 2-Dimensional Recording

A *page hologram* can be defined as a hologram made up of many nonoverlapping subholograms or pages [Rajchman 70]. A page is recorded through the use of a page composer, which is a device that stores data and converts it from electronic form to optical form, as indicated in Figure 54.

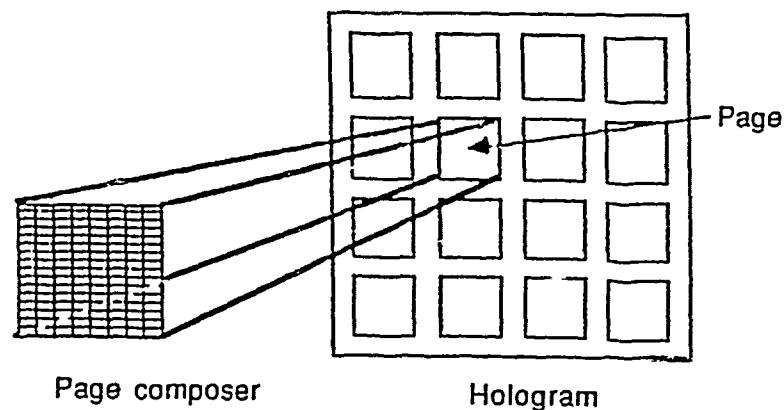


Fig 54. Page Hologram with Page Composer

To record a page, data is first loaded from the computer to the page composer, then a small area of the emulsion is exposed by an aperture, and the information stored in the page composer is flashed onto the exposed area. To record another page the aperture is moved, new data is loaded into the page composer and the process is repeated. Depending on the medium used, information can be erased from a page, and the page can be rewritten with new data.

Volume Holograms: 3-Dimensional Recording

If the recording medium is thick, a hologram may be classified as a *volume hologram* in which many wave patterns can be superimposed. Volume holograms have some very useful and interesting features which are distinct from those of plane holograms. Volume holograms lend themselves very nicely to the storage of information since many holograms are superimposed in the same recording volume. Each hologram is discernible from the others due to the fact that each hologram is recorded with a unique reference beam angle.

To reconstruct (read) a particular hologram the reference beam must be deflected to within a narrow angular corridor about the Bragg angle at which that particular hologram was recorded. Beam deflection for this purpose is accomplished by an acousto-optical modulator, or otherwise known as a Bragg Cell, which is discussed in the next section.

As the number of holograms recorded in a volume increases, the angular corridor decreases [Gaylord 79]. As early as 1968 it was shown that it was possible to superimpose 1000 holograms in the same photographic emulsion [LaMacchia 68].

The recording of more and more holograms in a particular volume, introduces the problem that a hologram being written will affect the holograms already present in the volume. So this is one of the factors, that limits the number of holograms that can be stored in a volume.

This particular problem was largely solved by applying an external electrical field to the material in which the recording is taking place, Lithium Niobate. The application of the electrical field, greatly increased the material's writing sensitivity, while its erasure sensitivity did not change. Thus, as a new hologram is written into the volume, only a slight erasure of the other holograms occurs.

Acousto-Optic Deflector (Bragg Cell)

A Bragg Cell is an acousto-optical deflector which allows the redirection of a light beam passing through it, as shown in Figure 55.

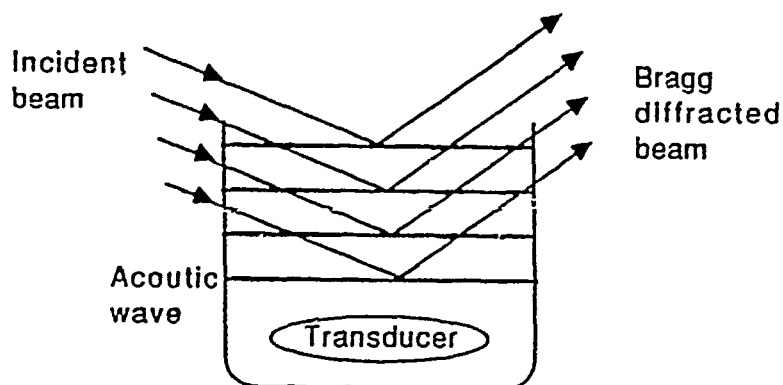


Fig 55. Acousto-Optical Deflector

At the bottom of the deflector cell there is an acoustic transducer. This transducer introduces an acoustic wave into the elasto-optic medium of the cell. In this way a periodic strain is established in the medium. The cell is now interpreted to be a stack of reflecting layers spaced by the acoustic wave's wavelength. By turning on and off and varying the acoustic wave's wavelength, the Bragg Cell can conveniently and quickly deflect the light beam passing through it.

Usually for holographic applications there will be a deflector for each dimension that the beam must be deflected in. For example, Figure 56 is a xy deflector system. Generalizing, a cascade combination of m deflectors allows 2^m deflection angles [Gaylord 79].

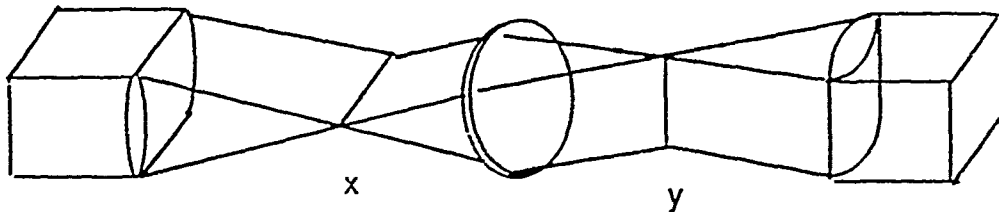


Fig 56. Page Addressing Deflector System

Magneto-Optic Holograms

Most holograms mentioned previously are write once holograms. A category of holograms that offers possible read write capabilities is magneto-optical holograms. The only difference between these and the holograms we have already described is the recording process. The storage medium is a thin film of MnBi, a magnetic material, that is magnetized normally to the surface into which holographic patterns are recorded by Curie-point writing, and read out is by the Kerr or Faraday effects [Mezrich 70]. These holograms, which have considerable update capability, could be useful in indexing data/knowledge bases that are not static.

Computer Generated Holograms

Computer Generated Holograms (CGH) are important because a hologram can be made of an object which does not physically exist. Such a hologram can be constructed by calculating the ideal wavefront on the basis of diffraction theory. The field on the hologram surface is represented by a transparency produced by a computer driven plotter, laser beam, or electron beam on correspondingly diverse materials [Tricoles 87]. The possibility here exists that holograms can be recorded close to real time.

DATABASE INDEX USING OCAMs

Sakaguchi demonstrated a relatively straight forward extension of conventional holographic techniques in order to develop a one dimensional optical content addressable memory [Sakaguchi 70]. Later [Knight 74] refined the idea into a page oriented holographic associative memory. Sakaguchi used two object beams $A(x)$, $B(x)$ and one reference beam, instead of one object and one reference beam to record a thin hologram. The two beams $A(x)$ and $B(x)$, contain the bit information, where $A = (A_1, A_2, A_3, \dots, A_n)$ and $B = (B_1, B_2, B_3, \dots, B_m)$. The A information is represented by beam $A(x)$, which consists of $2n$ coherent parallel beams, $a_1, \bar{a}_1, a_2, \bar{a}_2, a_3, \bar{a}_3, \dots, a_n, \bar{a}_n$, while the B information is represented by beam $B(x)$ consisting of $2m$ diverging beams $b_1, \bar{b}_1, b_2, \bar{b}_2, b_3, \bar{b}_3, \dots, b_m, \bar{b}_m$. Each bit of A and B is represented by a true beam a_i and b_i and a complement beam \bar{a}_i and \bar{b}_i , where i represents the i th bit.

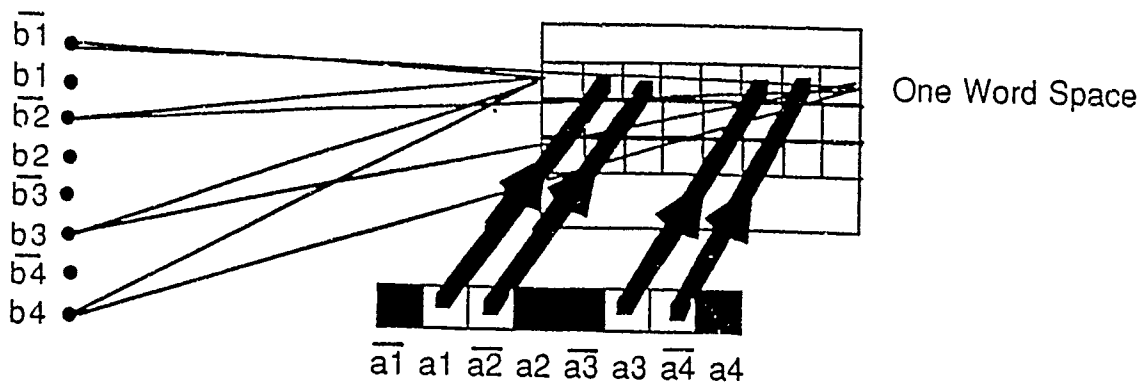


Fig 57. Sakaguchi's Associative Holographic Memory

The light beams that make up $A(x)$ and $B(x)$ come from the true or complement bit places, according to whether the corresponding bits are in states "1" or "0". As shown in Figure 57, suppose that (A_1, A_2, A_3, A_4) is (1,0,1,0) and (B_1, B_2, B_3, B_4) is (0,0,1,1). Then, $A(x)$ is a set of coherent beams coming from $a_1, \bar{a}_2, a_3, \bar{a}_4$, and $B(x)$ is a set of diverging beams coming from $\bar{b}_1, \bar{b}_2, b_3, b_4$. In this way the hologram is recorded with A being recorded as a spot sequence $a_1, \bar{a}_2, a_3, \bar{a}_4$ over the entire page in a word space, while B is contained in every spot of $A(x)$.

The holograms are read by interrogating them with a search argument $C = (C_1, C_2, C_3, \dots, C_n)$, represented by beam $C(x)$. Beam $C(x)$ consists of $2n$ coherent light beams, $c_1, \bar{c}_1, c_2, \bar{c}_2, c_3, \bar{c}_3, \dots, c_n, \bar{c}_n$. A match is detected by the Exclusive Or principle; that is a match occurs if the i th interrogation signal bit and the i th interrogated bit are different. So, in actuality the beam which does the interrogation, ($C'(x)$), is the complement of beam $C(x)$. The corresponding c_i or \bar{c}_i beams simultaneously illuminate the corresponding a_i or \bar{a}_i bits of all the words subjected to interrogation. Some of the c_i can be DON'T CARES, as such they do not participate in the interrogation so no light is emitted from the corresponding c_i or \bar{c}_i bit. To illustrate the point the previous values of A and B are used, where $A = (1,0,1,0)$ and $B = (0,0,1,1)$. If the search argument (C_1, C_2, C_3, C_4) is (1,0,dc,0), where dc denotes a DON'T CARE bit, the corresponding interrogation beam $C'(x)$ is actually (0,1,dc,1) that is, it consists of three parallel beams coming out of \bar{c}_1, c_2 and c_4 positions, with no light coming from c_3 or \bar{c}_3 , because the third bit is a don't care. We find that the interrogation signal $C = (1,0,dc,0)$ matches the interrogated signal $A = (1,0,1,0)$ in reality it is the EXOR between $C' = (0,1,dc,1)$ and $A = (1,0,1,0)$, which determines whether a match has occurred or not. The match is detected by a photodetector array; wherever the detector detects a null then a match has occurred. If all the bits for a word are detected as null then, the word matches the search argument.

Knight (1974) extended Sakaguchi's work by recording a page hologram consisting of 100 by 100 pages each containing 10^4 bits/page, for a total of 10^8 bits. Each page was very small; 1mm^2 . In recording the 10,000 pages the $A(x,y)$ and $B(x,y)$ beams were kept stationary while the reference beam was repositioned for each page. The page composer was loaded with new $A(x,y)$ and $B(x,y)$ data for each page.

What makes this suitable for implementing a database index is that each page has a code word, A , associated with it; and the data, B , written in a particular page can be identified by using the code word. In other words, the entire memory plane can be searched with a code word represented by beam $A(x,y)$ in parallel with one flash of light. When a match is detected by the photo-array, the data, represented by beam $B(x,y)$, corresponding to that page is read out. Next, an example will be presented that illustrates how this technique can be applied in implementing an

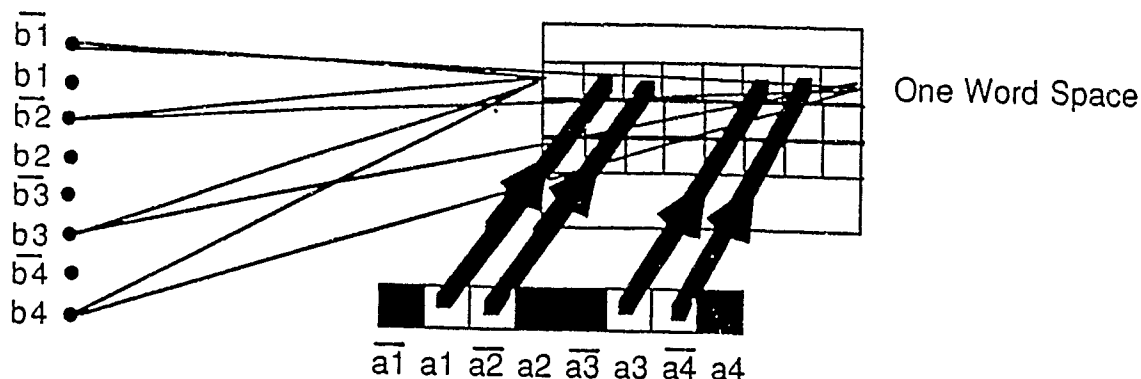


Fig 57. Sakaguchi's Associative Holographic Memory

The light beams that make up $A(x)$ and $B(x)$ come from the true or complement bit places, according to whether the corresponding bits are in states "1" or "0". As shown in Figure 57, suppose that (A_1, A_2, A_3, A_4) is (1,0,1,0) and (B_1, B_2, B_3, B_4) is (0,0,1,1). Then, $A(x)$ is a set of coherent beams coming from $a_1, \bar{a}_2, a_3, \bar{a}_4$, and $B(x)$ is a set of diverging beams coming from $\bar{b}_1, \bar{b}_2, b_3, b_4$. In this way the hologram is recorded with A being recorded as a spot sequence $a_1, \bar{a}_2, a_3, \bar{a}_4$ over the entire page in a word space, while B is contained in every spot of $A(x)$.

The holograms are read by interrogating them with a search argument $C = (C_1, C_2, C_3, \dots, C_n)$, represented by beam $C(x)$. Beam $C(x)$ consists of $2n$ coherent light beams, $c_1, \bar{c}_1, c_2, \bar{c}_2, c_3, \bar{c}_3, \dots, c_n, \bar{c}_n$. A match is detected by the Exclusive Or principle; that is a match occurs if the i th interrogation signal bit and the i th interrogated bit are different. So, in actuality the beam which does the interrogation, ($C'(x)$), is the complement of beam $C(x)$. The corresponding c_i or \bar{c}_i beams simultaneously illuminate the corresponding a_i or \bar{a}_i bits of all the words subjected to interrogation. Some of the c_i can be DON'T CARES, as such they do not participate in the interrogation so no light is emitted from the corresponding c_i or \bar{c}_i bit. To illustrate the point the previous values of A and B are used, where $A = (1,0,1,0)$ and $B = (0,0,1,1)$. If the search argument (C_1, C_2, C_3, C_4) is (1,0,dc,0), where dc denotes a DON'T CARE bit, the corresponding interrogation beam $C'(x)$ is actually (0,1,dc,1) that is, it consists of three parallel beams coming out of \bar{c}_1, c_2 and c_4 positions, with no light coming from c_3 or \bar{c}_3 , because the third bit is a don't care. We find that the interrogation signal $C = (1,0,dc,0)$ matches the interrogated signal $A = (1,0,1,0)$ in reality it is the EXOR between $C' = (0,1,dc,1)$ and $A = (1,0,1,0)$, which determines whether a match has occurred or not. The match is detected by a photodetector array; wherever the detector detects a null then a match has occurred. If all the bits for a word are detected as null then, the word matches the search argument.

Knight (1974) extended Sakaguchi's work by recording a page hologram consisting of 100 by 100 pages each containing 10^4 bits/page, for a total of 10^8 bits. Each page was very small; 1mm^2 . In recording the 10,000 pages the $A(x,y)$ and $B(x,y)$ beams were kept stationary while the reference beam was repositioned for each page. The page composer was loaded with new $A(x,y)$ and $B(x,y)$ data for each page.

What makes this suitable for implementing a database index is that each page has a code word, A , associated with it; and the data, B , written in a particular page can be identified by using the code word. In other words, the entire memory plane can be searched with a code word represented by beam $A(x,y)$ in parallel with one flash of light. When a match is detected by the photo-array, the data, represented by beam $B(x,y)$, corresponding to that page is read out. Next, an example will be presented that illustrates how this technique can be applied in implementing an

index scheme.

In this example we use an inverted list with indirect addressing as the index technique. This particular index was chosen over the more conventional inverted list index, due to the fact that whenever a reorganization of the database occurs (the physical addresses of the records change), all of the addresses in all the holograms would have to be changed. In conventional systems this is not an easy task, and in a medium such as holography it would be even more difficult. However, by using indirect addressing only the top level index would have to be changed.

Part of the example database is shown in Figure 58. The database is an employee database, and indexes are formed on SS#, Sex, Job, Dept# as shown in Figure 59.

Database						
SS#	Name	Sex	Job	Dept#	Salary	Seniority
012315678	Mary	F	SQ	120	1000	73
123456789	Gary	M	FI	110	2000	61
315678901	John	M	EN	110	1750	76
456789010	Peter	M	JN	130	900	84
567890123	Eva	F	FI	120	2400	79
678901234	Teresa	F	EN	120	1750	81
789012345	Sophie	F	JN	120	1000	83
890123456	Mark	M	SQ	130	1200	77
901234567	Laura	F	FI	110	2200	76
916801231	Hector	M	EN	130	2000	72
956801234	Bruce	M	EN	110	3000	70
968012345	Luke	M	FI	120	2400	74
979123456	Dean	M	SQ	130	1300	74
980123456	Sue	F	EN	120	2000	82
991245678	Cindy	F	JN	130	1100	85
999234567	Anna	F	JN	120	1200	83
999379135	Sam	M	EN	130	2300	80
999387913	Lucy	F	FI	110	2200	80
999581470	Ken	M	SQ	130	1300	72
999826048	Henry	M	FI	110	2400	70

Fig 58. Database Excerpt

A1 Index			A2 Index		A3 Index		A4 Index	
Serial#	SS#	Address	Sex	Serial#	Job	Serial#	Dept#	Serial#
1	012345678	32	M	2	SQ	1	110	2
2	123456789	23		3		8		3
3	345678901	11		4		13		9
4	456789010	43		8		19		11
5	567890123	16		10				18
6	678901234	10		11	FI	2	120	20
7	789012345	6		12		5		1
8	890123456	19		13		9		5
9	901234567	22		17		12		6
10	916801234	13		19		18		7
11	956801234	45		20		20		12
12	968012345	4	F	1	EN	3	130	14
13	979123456	12		5		6		16
14	980123456	21		6		10		4
15	991245678	30		7		11		8
16	999234567	36		9		14		10
17	999379135	40		14	JN	17		13
18	999387913	25		15		4		15
19	999581470	18		16		7		17
20	999826048	7		18		15		19
						16		

Fig 59. Indexes

The question now arises as to how we apply Knight's work to the above database index scheme. The pages represent the attributes we have indexed on. For example looking at the Department index we see that there are 3 different departments (110, 120, 130). With each department number we have associated a list of serial #'s. In recording the data we place the department number on the A(x,y) beam and the list of serial #'s associated with that particular department number on the B(x,y) beam. The representation is stored as shown in Figure 60.

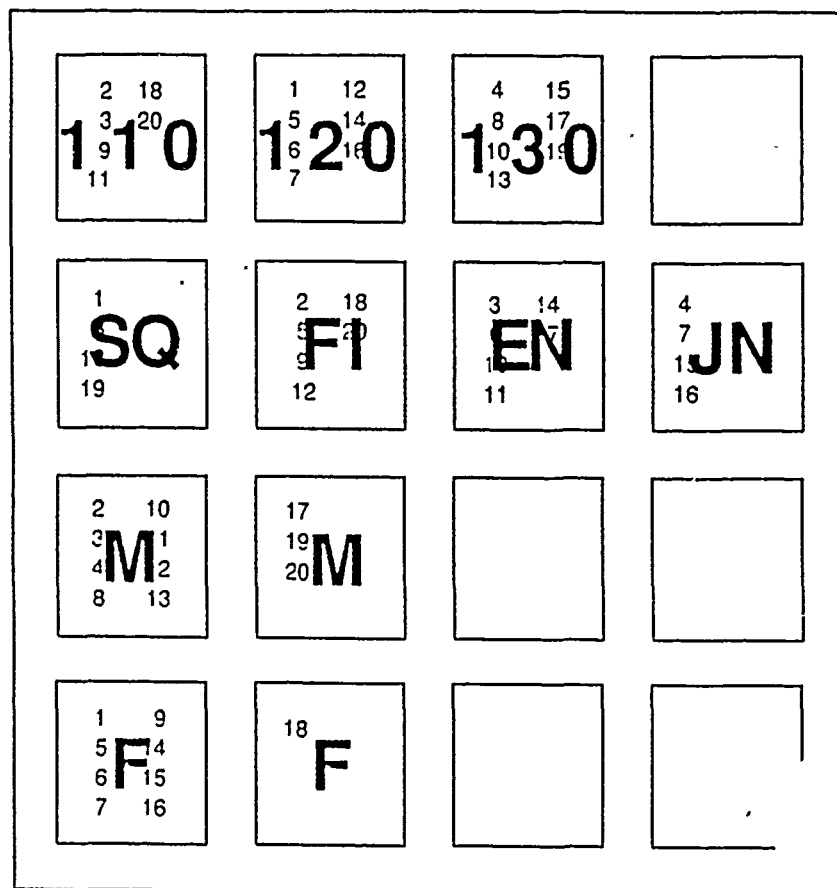


Fig 60. OCAM Representation

In order to illustrate how queries are performed suppose information is needed on those persons working in department 120. In this case "120" is placed on beam C(x,y), the interrogation beam. The interrogation signal is flashed onto the memory plane, and the match is detected by the photo-detector array. The contents of the page are thus read in parallel and at the speed of light. The situation may exist that the data concerning department 120 may span several pages. In that case each of the pages must be read in turn to get the serial #'s. The serial #'s are now used to obtain the data records.

CONCLUSION

In this paper we have tried to illustrate how optical content addressable memories can be used in the management of indexes to very large data/knowledge bases. OCAMs have many attributes, large size, almost instantaneous access and parallel readout, that are important to solving the problems associated with managing an index for very large data/knowledge bases. However, it does take a relatively long time to form a hologram and they are difficult to change once formed. We are continuing our research into many of the issues concerning this marriage and will be reporting on our work from time to time.

REFERENCES

- [CAULFIELD70] H.J. Caulfield and Sun Lu, *The Applications of Holography*, Wiley-Interscience, Division of John Wiley & Sons, 1970
- [COLLIER71] R.J. Collier, C.B. Burckhardt, L.H. Lin, *Optical Holography*, Academic Press Inc., 1971
- [GABOR69] D. Gabor, "Associative Holographic Memories", IBM Research Journal, March 1969
- [GAYLORD79] Thomas K. Gaylord, *Handbook of Optical Holography*, Chapter on Applications, Academic Press Inc. 1979
- [KNIGHT74] Gordon R. Knight, "Page-Oriented Associative Holographic Memory", Applied Optics, Vol. 13, No. 4, April 1974
- [LAMACCHIA68] J.T. LaMacchia and D.L. White, "Coded Multiple Exposure Holograms", Applied Optics, Vol. 7, No. 1, January 1968.
- [MEZRICH70] R.S. Mezrich, "Magnetic Holography", Applied Optics, Vol. 9, No. 10, October 1970
- [RAJCHMAN70] J.A. Rajchman, "An Optical Read-Write Mass Memory", Applied Optics, Vol. 9, No. 10, October 1970
- [SAKAGUCHI70] M. Sakaguchi, N. Nishida, T. Nemoto, "A New Associative Memory System Utilizing Holography", IEEE Transactions on Computers, Vol. C-19, No. 12, December 1970
- [SMITH69] Howard M. Smith, *Principles of Holography*, Wiley-Interscience 1969
- [TRICOLES87] G. Tricoles, "Computer Generated Holograms: a historical overview", Applied Optics, Vol. 25, No. 20, October 1987